Prof. Dr. Dimitri Kuzmin
Dipl.-Math. Justus Klipstein
Numerische Mathematik für Physiker und Ingenieure

Abgabe der **Aufgaben bis Donnerstag, 07.06.2018, 16:15 Uhr vor der Vorlesung** (einzeln, zu zweit oder dritt) in den Briefkasten Ihrer Übungsgruppe:

Gruppe	Termin	Übungsleiter	Briefkasten
2	Montag 10:00-12:00	Korinna Rosin	107
3	Montag 12:00-14:00	Mirco Arndt	108
4	Montag 14:00-16:00	Marina Bangert	109
5	Montag 16:00-18:00	Marina Bangert	109
6	Dienstag 14:00-16:00	Justus Klipstein	108
7	Dienstag 16:00-18:00	Dr. Fatma Ibrahim	110

Die Briefkästen befinden sich im Foyer des Mathematikgebäudes. Bitte vermerken Sie unbedingt auf jeder Abgabe Ihren Namen und Ihre Übungsgruppe. Eine Abgabe der theoretischen Übungsaufgaben per E-Mail ist nicht möglich.
Abgabe der **Programmieraufgaben bis Donnerstag, 07.06.2018, 24:00 Uhr** digital im Moodle-Arbeitsraum der Veranstaltung. Bei Abgabe zu zweit oder dritt bitte nur einmal einreichen und im Kommentar den Namen und E-Mail Adresse des/der Koautors/Koautorin/Koautoren nennen. Sie können die Aufgaben in MatLab bzw. Octave schreiben. Bitte den "Header" als Kommentar in die Programme einfügen.

Aufgabe 5.1 (Quadraturformeln | 4 + 2 + 1 = 7 Punkte)

a) Es seien $0 < c < 1$ und drei Stützstellen mit zugehörigem Funktionswert gegeben $(-c, f(-c))$, $(0, f(0))$, $(c, f(c))$. Legen Sie durch diese drei Punkte ein Polynom

$$p_2(x) = a_0 + a_1 x + a_2 x^2 \in \mathbb{P}_2.$$

a) Entwickeln Sie daraus eine Quadraturformel $I_q(f)$ auf dem Intervall $[-1, 1]$. Wie ist c zu wählen, damit **2** Funktionsauswertungen genügen, um jedes Polynom bis zum Polynomgrad zwei exakt zu integrieren?
Tipp: Beginnen Sie damit die Koeffizienten von $p_2(x)$ aus den Interpolationsbedingungen zu bestimmen.

b) Ist die gefundene Quadraturformel auch noch für Polynome vom Grad drei exakt?

c) Welche aus der Vorlesung bekannte Quadraturformel ergibt sich für $c = \sqrt{\frac{3}{5}}$.

Aufgabe 5.2 (LR- und Cholesky-Zerlegung | 4+1+3 Punkte)

a) Bestimmen Sie die LR-Zerlegung (ohne Pivotisierung) der Matrix

$$A = \begin{pmatrix} 9 & -6 & 0 & -3 \\ -6 & 20 & 8 & 2 \\ 0 & 8 & 20 & 4 \\ -3 & 2 & 4 & 6 \end{pmatrix}$$

b) Berechnen Sie mittels der LR-Zerlegung die Determinante der Matrix A

c) Bestimmen Sie aus der LR-Zerlegung die Cholesky Zerlegung von A in eine untere Dreiecksmatrix $\tilde{L}$ mit

$$A = \tilde{L}\tilde{L}^\top$$

Aufgabe 5.3 (Thomas-Algorithmus | 5 Punkte)

Bestimmen Sie mit Hilfe des Thomas-Algorithmus (Gauss-Elimination für Tridiagonalmatrizen) die LR-Zerlegung der Tridiagonalmatrix

$$A = \begin{bmatrix} 2 & 1 & 0 & 0 & 0 & 0 \\ 1 & 2 & 1 & 0 & 0 & 0 \\ 0 & 1 & 2 & 1 & 0 & 0 \\ 0 & 0 & 1 & 2 & 1 & 0 \\ 0 & 0 & 0 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 & 1 & 2 \end{bmatrix}.$$

Programmieraufgabe 5.1 (LR-Zerlegung mit dem Thomas-Algorithmus | 14 Punkte)

Implementieren Sie die LR-Zerlegung mit dem **Thomas-Algorithmus (ohne Pivotisierung)** mit Vorwärts- und Rückwärtssubstitution. Schreiben Sie dazu die folgenden Routinen, ohne dabei entsprechende Funktionen in Matlab zu nutzen (wie etwa den Backslash-Löser \):

a) `function LR = thomas_decompose(A)`
Diese Funktion soll die tridiagonal Matrix $A \in \mathbb{R}^{n\times n}$ in $A = L*R$ zerlegen, wobei L eine untere Dreiecksmatrix ist, bei der die Diagonaleinträge immer 1 sind und R eine obere Dreiecksmatrix ist. Bei L und R sind jeweils nur die Hauptdiagonale und die untere bzw. obere Nebendiagonale besetzt. Die Matrizen L und R sollen, wie in der Vorlesung vorgestellt platzsparend in **einer einzigen** tridiagonal Matrix namens LR gespeichert werden (also ohne eine explizite Speicherung der Einsen auf der Diagonale von L). Verwenden sie das MatLab/Octave Matrixformat "sparse(n)" um die Matrizen als $n \times n$ Bandmatrize zu speichern oder alternativ drei Vektoren der länge n als $n \times 3$-Matrix.
Hinweis: Eine leere Sparse-Matrix mit Speicherplatz für drei Bänder lässt sich mit `A = spalloc(n,n,3*n)` erzeugen.

b) `function y = forward_solve(LR,b)`
Diese Funktion soll das Gleichungssystem $Ly = b$ durch Vorwärtssubstitution lösen, wobei LR die oben beschriebene Matrix aus der LR-Zerlegung ist.

`function x = backward_solve(LR,y)`
Diese Funktion soll das Gleichungssystem $Rx = y$ durch Rückwartssubstitution lösen, wobei LR die oben beschriebene Matrix aus der LR-Zerlegung ist.

c) `function LR_Test()`
Testen Sie in dieser Funktion Ihre Implementierung aus Teilen **a)** und **b)** jeweils für $n = 10000, 100000, 1000000, 10000000$ für das lineare $n \times n$ Gleichungssystem

$$Ax = b$$

mit der Matrix "`A = gallery('tridiag',n, -2, 8, -2);`" (8 auf der Hauptdiagonalen und -2 auf den Nebendiagonalen), sowie der rechten Seite $b_i = 1$ für $i < \frac{n}{2}$ und $b_i = 2$ sonst. Geben Sie insbesondere für jedes n das Residuum $\|Ax - b\|_2$ an.
Speichern Sie mit "`tic; ... t(i) = toc;`" die Zeiten die zum bestimmen der LR-Zerlegung mit dem Thomas-Algorithmus gebraucht werden.

d) Entspricht das Wachstum der Rechenzeit den Angaben in der Vorlesung. Mit welchem α steigt die Rechenzeit in der Schreibweise $\mathbf{O}(n^\alpha)$. Vergleichen sie die Rechenzeit mit der in moodle zu Verfügung stehenden "LR-Zerlegung.zip".
Hinweis: Verwenden Sie nicht die zu Verfügung stehenden `function x = backward_solve (LR,y)` und `function y = forward_solve(LR,b)` für den Thomas-Algrithmus ohne sie anzupassen.