Übungsblatt 3

Lösung großer linearer Gleichungssysteme

Aufgabe 1. *LUP-Faktorisierung*
Gegeben sei die Matrix

$$A = \begin{pmatrix} 0 & 5 & 0 & 4 \\ -1 & -6 & 0 & -2 \\ 9 & 3 & -2 & 1 \\ -4 & 7 & -1 & 3 \end{pmatrix}.$$

(a) Schreiben Sie ein Programm zur Berechnung der LUP-Faktorisierung mit Spaltenpivotsuche. Die Berechnung der einzelnen Elemente von L, U und P folgt aus der Rechenvorschrift:

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} & \dots \\ a_{21} & a_{22} & a_{23} & \dots \\ a_{31} & a_{32} & a_{33} & \dots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix} = LUP = \begin{pmatrix} 1 & 0 & \dots & \dots \\ l_{21} & 1 & 0 & \ddots \\ l_{31} & l_{32} & 1 & 0 \\ \vdots & \dots & \ddots & \ddots \end{pmatrix} \begin{pmatrix} u_{11} & u_{12} & \dots & \dots \\ 0 & u_{22} & u_{23} & \dots \\ \vdots & \ddots & u_{33} & u_{34} \\ \vdots & \dots & \ddots & \ddots \end{pmatrix} P$$

$$= \begin{pmatrix} u_{11} & u_{12} & u_{13} & u_{14} & \dots \\ l_{21}u_{11} & l_{21}u_{12} + u_{22} & l_{21}u_{13} + u_{23} & \dots & \dots \\ l_{31}u_{11} & l_{31}u_{12} + l_{32}u_{22} & l_{31}u_{13} + l_{32}u_{23} + u_{33} & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{pmatrix} P,$$

wobei P eine *Permutationsmatrix* ist, die zur Wahl der Pivotelemente benötigt wird. Die Matrix L ist eine sogenannte *untere Dreiecksmatrix*, U ist eine *obere Dreiecksmatrix*. Da P eine orthogonale Matrix ist, gilt $P^{-1} = P^{\mathsf{T}}$. Daraus folgt:

$$AP^{\mathsf{T}} = LU. \tag{1}$$

Durch Koeffizientenvergleich folgt der Algorithmus für die LUP-Zerlegung:

$$\begin{aligned} l_{ii} &= 1, \\ l_{ij} &= \frac{1}{u_{jj}} \left(a^*_{ij} - \sum_{k=1}^{j-1} l_{ik} u_{kj} \right) && (j < i), \\ u_{ij} &= a^*_{ij} - \sum_{k=1}^{i-1} l_{ik} u_{kj} && (j \geq i). \end{aligned}$$

Hier bezeichnet a^*_{ij} die Komponente $(AP^{\mathsf{T}})_{ij}$. Die Koeffizienten werden mit zwei Schleifen (Zählvariablen i, j) durchlaufen, wobei in der äußeren Schleife i inkrementiert wird, in der Inneren j. Vor jeder Inkrementierung von i, das heißt am Ende der äußeren Schleife, muss überprüft werden, ob eine Spaltenvertauschung vorgenommen werden

muss. Dies ist genau dann notwendig, wenn das Diagonalelement u_{ii} betragsmäßig klein oder gar Null ist. Dann müssen Spalten von A (und somit auch von U) vertauscht werden, um zu garantieren, dass im Folgeschritt kein ungünstiger Koeffizient (ungünstig wegen Rundungsfehler) auftreten kann. Bei jeder Spaltenvertauschung ändert sich die Permutationsmatrix P, die Matrix A^* und U.

Um Speicherplatz zu sparen sollen L und U in einer gemeinsamen Matrix B gespeichert werden. Dafür müssen in den obigen Gleichungen lediglich die Ausdrücke l_{ij}, u_{ij} durch b_{ij} ersetzt und der Ausdruck $l_{ii} = 1$ gestrichen werden. Letzteres ist problemlos möglich, da die Diagonalelemente von L per Definition 1 sind. Die Matrix B hat also die Struktur

$$B = \begin{pmatrix} u_{11} & u_{12} & u_{13} & \dots \\ l_{21} & u_{22} & u_{23} & \dots \\ l_{31} & l_{32} & u_{33} & \dots \\ \vdots & \ddots & \ddots & \dots \end{pmatrix}.$$

Die Spaltenvertauschung wird dann auf die Matrix B angewendet (in der Vorlage entspricht B der Matrix `LU`). Verwenden Sie die ausführlich dokumentierte Vorlage. Testen Sie den Algorithmus mit einigen Beispielen. Dabei sollten Sie auch solche Matrizen wählen, die Spaltenvertauschungen *zwingend* benötigen, beispielsweise eine Permutationsmatrix der Form

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

(b) Schreiben Sie eine Subroutine zur Lösung des linearen Gleichungssystem $Ax = y$. Dabei soll die berechnete LUP-Faktorisierung verwendet werden. Die Lösung des linearen Gleichungssystems erfolgt in drei Schritten:

$$LUPx = L\underbrace{(UPx)}_{=z} = \begin{pmatrix} 1 & 0 & \dots & \dots \\ l_{21} & 1 & \ddots & \vdots \\ l_{31} & l_{32} & 1 & \ddots \\ \vdots & \vdots & \ddots & \ddots \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ \vdots \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \end{pmatrix} \text{ Vorwärtseinsetzen } \Longrightarrow z,$$

$$U\underbrace{Px}_{=v} = \begin{pmatrix} u_{11} & u_{12} & \dots & \dots \\ 0 & u_{22} & u_{23} & \ddots & \vdots \\ \vdots & \ddots & u_{33} & \ddots & \ddots \\ \vdots & \vdots & \ddots & \ddots & \ddots \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ \vdots \end{pmatrix} = \begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ \vdots \end{pmatrix} \text{ Rückwärtseinsetzen } \Longrightarrow v,$$

$$v = Px \qquad \text{Permutation } \Longrightarrow x = P^{\mathsf{T}}v.$$

Implementieren Sie eine automatische Fehlerkorrektur nach folgendem Schema:

[S1] Initialisiere $j = 0$, $x = 0$, $r = -y$; setze $N_{\max}$ und $\varepsilon_{\mathrm{TOL}}$;

[S2] Falls $\dfrac{\|r\|_2}{\|y\|_2} < \varepsilon_{\mathrm{TOL}}$: *OK, ENDE*;

[S3] Löse $A\delta x + r = 0$; Aktualisiere $x = x + \delta x;\ r = Ax - y$;

[S4] Inkrementiere j; Falls $j \leq N_{\max}$: gehe zu **[S2]**.

Testen Sie das Programm an einem Zufallsvektor y.

(c) Ersetzen Sie nun die Matrix A durch eine Zufallsmatrix und berechnen Sie das Residuum $\|LUP - A\|_F$. Testen Sie das Programm mit einer zufälligen rechten Seite y und überprüfen Sie die Residuen $\|Ax - y\|_1$, $\|Ax - y\|_2$ und $\|Ax - y\|_\infty$. Variieren Sie die Dimension von A (z.B. 10, 20, 50, 200, 500, 1000). Wie verändert sich das Residuum durch die Fehlerkorrektur? Variieren Sie die Anzahl der zulässigen Korrekturschritte $N_{\max}$.

Aufgabe 2. *Iterative Gleichungslöser*

Importieren Sie zunächst aus der Datei `sparse_matrix.dat` (ILIAS) eine dünnbesetzte, großdimensionale Bandmatrix B_0. Berechnen Sie

$$B = B_0 {B_0}^{\mathsf{T}}.$$

(a) Visualisieren Sie die Bandstruktur der Matrix B mit Hilfe des Befehls `spy`.

(b) Programmieren Sie den in der Vorlesung vorgestellten Gauss-Seidel Algorithmus (siehe auch `http://de.wikipedia.org/wiki/Gauß-Seidel-Verfahren`). Modifizieren Sie die Matrix B, so dass die Diagonalelemente mit einem Faktor k multipliziert werden. Testen Sie den Algorithmus bei Verwendung einer zufälligen rechten Seite und zunächst für $k = 2$. Wie viele Iterationen werden benötigt, bis

$$\frac{\left\|Bx^{(j)} - y\right\|_2}{\|y\|_2} < 10^{-8}$$

erreicht wird? Verwenden Sie verschiedene rechte Seiten y. Wie groß ist der Einfluss der rechten Seiten des Gleichungssystems auf die Anzahl der Iterationen?

(c) Reduzieren Sie die Diagonaldominanz von B, indem Sie den Faktor k reduzieren. Was beobachten Sie für die Anzahl der notwendigen Iterationen? Was folgern Sie daraus für die Anwendbarkeit des Algorithmus?

(d) Verwenden Sie nun die von Matlab zur Verfügung gestellten Gleichungslöser. Es ist unbedingt ratsam einen Vorkonditionierer zu verwenden. Dafür geben Sie eine dünnbesetzte Matrix D vor, die aus den Diagonalelementen von B besteht:

$$D = \mathrm{diag}(b_{11},\, b_{22},\, \ldots, b_{NN}).$$

Es wird dann statt $Bx = y$ das System $D^{-1}Bx = D^{-1}y$ gelöst. Die Matrix D muss hierfür einfach (d.h. numerisch kostengünstig) invertierbar sein und B^{-1} möglichst gut approximieren. Die Aufrufe der verschiedenen Gleichungslöser sind in der Matlab-Hilfe dokumentiert (siehe ”iterative methods - sparse systems “). Die Wahl einer regulären Diagonalmatrix benötigt bei der Inversion (fast) keine numerischen Kosten, ist jedoch nicht immer optimal. Es existieren weitere Möglichkeiten der Vorkonditionierung (incomplete LU factorization, Vorkonditionierung mit Dreiecksmatrix, . . .).

[Testat 1] - Abgabe per Upload auf ILIAS bis zum 09.11.2017

Cholesky-Zerlegung.

Gegeben ist die symmetrische Matrix

$$A = \begin{pmatrix} 4 & 2 & -1 & 0 & 0 & 0 \\ & 4 & 1 & 1 & 0 & 1 \\ & & 5 & 3 & -1 & 2 \\ & & & 4 & 1.1 & 2.5 \\ & & \text{sym.} & & 2.4 & 1 \\ & & & & & 3 \end{pmatrix}.$$

(a) Zeigen Sie, dass die Matrix A positiv definit ist.

(b) Schreiben Sie ein Programm zur Lösung des linearen Gleichungssystems $Ax = y$. Verwenden Sie eine eigene Implementierung der Cholesky-Zerlegung (s. Vorlage).

Wenden Sie Ihre Routine an, um die Choleksy-Zerlegung von A zu berechnen, und lassen Sie die Matrix L ausgeben. Überprüfen Sie die Zerlegung, indem Sie die Frobenius-Norm (`norm(A,'fro')`) des Residuums $R = A - LL^{\mathsf{T}}$ auswerten und auf der Konsole ausgeben lassen.

(c) Lösen Sie das Gleichungssystem

$$Ax = LL^T x = y$$

durch Vorwärts-Rückwärtseinsetzen:

- Bestimmen Sie z, so dass $Lz = y$ gilt. Starten Sie mit $z_1 = y_1/L_{1,1}$ und verwenden Sie die Rekursionsvorschrift $z_{n+1} = (y_{n+1} - \sum_{i=1}^{n} L_{n+1,i} z_i)/L_{n+1,n+1}$.

- Lösen Sie nun auf ähnliche Weise $L^T x = z$ durch Rückwärtseinsetzen (Start mit der Berechnung von x_N, umgekehrte Rekusion).

(d) Testen Sie Ihr Programm mit der rechten Seite $y = (1, -1, 5, 7, 6, -3)^{\mathsf{T}}$. Berechnen Sie das Residuum $r = Ax - y$, sowie $\|r\|_1$, $\|r\|_2$ und $\|r\|_\infty$.

Kontakt

Dipl.-Ing. Johannes Ruck johannes.ruck@kit.edu
M.Sc. Hannes Erdle hannes.erdle@kit.edu
Sprechstunde Do. 13:00-14:00 Uhr (Geb. 10.23, Raum 302.3)