Numerische Lineare Algebra
Spezielle Systeme

Friedrich Solowjow

2. Mai 2012, Bonn

Gliederung

Einleitung

Viele numerische Probleme führen auf lineare Gleichungssysteme.
Diese müssen unter folgenden Gesichtspunkten gelöst werden:

- Schnelligkeit
- Genauigkeit
- Effiziente Speichernutzung

Vorteile durch Strukturausnutzung von dünnbesetzten Gleichungssystemen:

- Betrachtung der Nicht-Null-Einträge
- Rechenzeiteinsparung
- Speicherplatzeinsparung

Spezielle Systeme

Es werden u. a. folgende Matrixstrukturen unterschieden:

- Definitheit
- Symmetrie
- Blocksysteme
- Vandermonde
- Tridiagonalmatrix
- Obere Hessenbergmatrix
- Bandstruktur

Definitheit

Definition

Sei $A \in \mathbb{R}^{nxn}, x \in \mathbb{R}^n$. A ist:

- positiv definit, falls $x^T * A * x > 0$
- positiv semidefinit, falls $x^T * A * x \geq 0$
- negativ definit, falls $x^T * A * x < 0$
- negativ semidefinit, falls $x^T * A * x \leq 0$
- indefinit, sonst

Symmetrie

Definition

Sei $A \in \mathbb{R}^{mxn}$. A heißt symmetrisch, wenn $A = A^T$.

$$\text{Beispiel: } A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix} = \begin{pmatrix} a_{11} & \cdots & a_{m1} \\ \vdots & \ddots & \vdots \\ a_{1n} & \cdots & a_{mn} \end{pmatrix} = A^T$$

Blockmatrizen

Definition

Eine Blockmatrix, ist eine Matrix, die sich durch ein System kleinerer Matrizen darstellen lässt.

Beispiel: Sei $A \in \mathbb{R}^{2nx2n}$ und $B, C, D, E \in \mathbb{R}^{nxn}$: $A = \begin{pmatrix} B & C \\ D & E \end{pmatrix}$

Vandermonde-Matrix

Definition

Eine Vandermonde-Matrix wird durch die folgende Form definiert:

$$V_n = \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ x_1 & x_2 & x_3 & \dots & x_n \\ x_1^2 & x_2^2 & x_3^2 & \dots & x_n^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_1^{n-1} & x_2^{n-1} & x_3^{n-1} & \dots & x_n^{n-1} \end{pmatrix}$$

Tridiagonalmatrix

Definition

Sei $A \in \mathbb{R}^{nxn}$.A heißt Tridiagonalmatrix, wenn
für $i > j+1$ und $j > i+1$ alle $a_{ij} = 0$.

$$\text{Beispiel: } A = \begin{pmatrix} a_{11} & a_{12} & 0 & \cdots & 0 \\ a_{21} & a_{22} & a_{23} & \ddots & \vdots \\ 0 & a_{32} & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & a_{(n-1)n} \\ 0 & \cdots & 0 & a_{n(n-1)} & a_{nn} \end{pmatrix}$$

Obere Hessenbergmatrix

Definition

Sei $H \in \mathbb{R}^{nxn}$. A heißt (obere) Hessenberg-Matrix, wenn $h_{i,j} = 0$, für alle $i > j + 1$.

$$\text{Beispiel: } H = \begin{pmatrix} h_{11} & h_{12} & h_{13} & \dots & h_{1n} \\ h_{21} & h_{22} & h_{23} & \dots & h_{2n} \\ 0 & h_{32} & h_{33} & \dots & h_{3n} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & h_{n(n-1)} & h_{nn} \end{pmatrix}$$

Bandmatrix

Definition

Seien $p, q \in \mathbb{N}$ mit $p, q \geq 0$ und $A \in \mathbb{R}^{nxn}$. A ist genau dann eine Bandmatrix der Bandbreite $l = p + q + 1$, falls alle $a_{ij} = 0$, für $j + p < i$ oder $i + q < j$.

Bandmatrix

$$A = \begin{pmatrix}
a_{11} & \cdots & a_{1(q+1)} & 0 & \cdots & \cdots & \cdots & 0 \\
\vdots & \ddots & & \ddots & \ddots & & & \vdots \\
a_{(p+1)1} & & \ddots & & \ddots & \ddots & & \vdots \\
0 & \ddots & & \ddots & & \ddots & \ddots & \vdots \\
\vdots & \ddots & \ddots & & \ddots & & \ddots & 0 \\
\vdots & & \ddots & \ddots & & \ddots & & a_{(n-q)n} \\
\vdots & & & \ddots & \ddots & & \ddots & \vdots \\
0 & \cdots & \cdots & \cdots & 0 & a_{n(n-p)} & \cdots & a_{nn}
\end{pmatrix}$$

Gliederung

Band-LU-Zerlegung

Satz 1

Sei $A \in \mathbb{R}^{nxn}$ mit einer LU Zerlegung. Wenn A die obere Bandbreite q und die untere Bandbreite p hat, dann hat U die obere Bandbreite q und L hat die untere Bandbreite p.

Beweis

Der Beweis folgt induktiv. Aus der LU-Zerlegung von A ergibt sich:

$$A = \begin{pmatrix} \alpha & \omega^T \\ \upsilon & B \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ \upsilon/\alpha & I_{n-1} \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & B - \upsilon\omega^T/\alpha \end{pmatrix} \begin{pmatrix} \alpha & \omega^T \\ 0 & I_{n-1} \end{pmatrix}$$

Mit den Eigenschaften der Teilmatrizen und der Induktionsvoraussetzung folgt:

$$L = \begin{pmatrix} 1 & 0 \\ \upsilon/\alpha & L_1 \end{pmatrix} \text{ und } U = \begin{pmatrix} \alpha & \omega^T \\ 0 & U_1 \end{pmatrix}$$

Band-Gauss-Eliminationsverfahren

Idee

Sei $A \in \mathbb{R}^{nxn}$ mit oberer Bandbreite q und unterer Bandbreite p.
Es wird eine Lösung x des Problems $Ax = b$ gesucht.
Der folgende Algorithmus führt eine LU-Zerlegung durch und überschreibt $A(i,j)$ mit

- $L(i,j)$, falls $i > j$
- $U(i,j)$, falls $i \leq j$

Pseudocode Band-Gauss-Eliminationsverfahren

for $k = 1{:}n - 1$ **do**
  for $i = k + 1{:}min(k + p, n)$ **do**
    $A(i, k) = A(i, k)/A(k, k)$
  end for
  for $j = k + 1{:}min(k + q, n)$ **do**
    for $i = k + 1{:}min(k + p, n)$ **do**
      $A(i, j) = A(i, j) - A(i, k)A(k, j)$
    end for
  end for
end for

Wenn $n >> p$ und $n >> q$, dann braucht der Algorithmus $2npq$ Flops.

Band-Vorwärtssubstitution

Es wird eine Lösung x zu dem Problem $Lx = b$ gesucht.

Idee

Sei $L \in \mathbb{R}^{nxn}$ eine untere Dreiecksmatrix mit Bandbreite p. Der folgende Algorithmus überschreibt b mit der Lösung des Problems $Lx = b$.

for $j = 1{:}n$ **do**
  for $i = j + 1{:}min(j + p, n)$ **do**
    $b(i) = b(i) - L(i, j)b(j)$
  end for
end for

Wenn $n >> p$ und $n >> q$, dann braucht der Algorithmus $2np$ Flops.

Band-Rückwärtssubstitution

Es wird eine Lösung x zu dem Problem $Ux = b$ gesucht.

Idee

Sei $U \in \mathbb{R}^{nxn}$ eine obere Dreiecksmatrix mit Bandbreite q. Der folgende Algorithmus überschreibt b mit der Lösung des Problems $Ux = b$.

for $j = n{:}-1{:}1$ **do**
  $b(j) = b(j)/U(j,j)$
  for $i = max(1, j-q){:}j-1$ **do**
    $b(i) = b(i) - U(i,j)b(j)$
  end for
end for

Wenn $n >> p$ und $n >> q$, dann braucht der Algorithmus $2nq$ Flops.

Band-Gauss-Eliminationsverfahren mit Pivotisierung

Satz 2

Sei $A \in \mathbb{R}^{nxn}$ eine nichtsinguläre Bandmatrix, mit oberer Bandbreite p und unterer Bandbreite q. Wenn man das Gaußsche Eliminationsverfahren mit partieller Pivotisierung benutzt um die Gauß-Transformation

$$M_j = I - \alpha^{(j)} e_j^T, \qquad j = 1{:}n-1$$

und die Permutationsmatrizen $P_1, \ldots, P_{n-1}$ zu berechnen, sodass $M_{n-1}P_{n-1}\ldots M_1 P_1 A = U$ obere Dreiecksmatrix ist, dann gelten folgende Zusammenhänge:

1. U hat obere Bandbreite $p + q$
2. $\alpha^{(j)} = 0$, wenn $i \leq j$ oder $i > j + p$

Obere Hessenbergmatrix

Definition

Sei $H \in \mathbb{R}^{nxn}$. A heißt (obere)Hessenberg-Matrix, wenn $h_{i,j} = 0$, für alle $i > j + 1$.

$$\text{Beispiel: } H = \begin{pmatrix} h_{11} & h_{12} & h_{13} & \dots & h_{1n} \\ h_{21} & h_{22} & h_{23} & \dots & h_{2n} \\ 0 & h_{32} & h_{33} & \dots & h_{3n} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & h_{n(n-1)} & h_{nn} \end{pmatrix}$$

Hessenberg-LU

Idee

Sei $H \in \mathbb{R}^{nxn}$ eine obere Hessenbergmatrix, $U \in \mathbb{R}^{nxn}$ eine obere Dreiecksmatrix und $M_i, P_i \in \mathbb{R}^{nxn}$ für $1 \leq i < n$.
Gaußtransformation: $M_{n-1} P_{n-1} \dots M_1 P_1 H = U$.

- Wenn $i \leq k$ $\quad H(i,k) = U(i,k)$
- Wenn $i = k+1$ $\quad H(i,k) = (M_k)_{k+1,k}$

piv($1{:}n-1$) ist ein Vektor, der Permutationen kodiert.

Pseudocode Hessenberg-LU

for $k = 1{:}n-1$
    if $|H(k,k)| < |H(k+1,k)|$
        $piv(k) = 1; H(k,k{:}n) \leftrightarrow H(k+1,k{:}n)$
    else
        $\mathrm{piv}(k) = 0$
    end
    if $H(k,k) \neq 0$
        $t = -H(k+1,k)/H(k,k)$
        for $j = k+1{:}n$
            $H(k+1,j) = H(k+1,j) + tH(k,j)$
        end
        $H(k+1,k) = t$
    end
end

Der Algorithmus braucht n^2 Flops.

Band-Cholesky

Idee

Sei $A \in \mathbb{R}^{nxn}$ eine symmetrische und positiv definite Matrix. Aus der Cholesky-Zerlegung folgt: $A = GG^T$ $(G \in \mathbb{R}^{nxn})$.
Aus Satz 1 folgt: G hat die selbe Bandbreite wie A.

Pseudocode Band-Cholesky

for $j = 1{:}n$
    for $k = \max(1, j-p){:}j-1$
        $\lambda = \min(k+p, n)$
        $A(j{:}\lambda, j) = A(j{:}\lambda, j) - A(j,k)A(A(j{:}\lambda, k))$
    end
    $\lambda = \min(j+p, n)$
    $A(j{:}\lambda, j) = A(j{:}\lambda, j)/\sqrt{A(j,j)}$
end

Wenn $n >> p$, dann braucht der Algorithmus $n(p^2 + 3p)$ Flops und n Wurzeln.

Tridiagonal-Systeme

Definition

Sei $A \in \mathbb{R}^{nxn}$.A heißt Tridiagonalmatrix, wenn für $i > j+1$ und $j > i+1$ alle $a_{ij} = 0$.

$$\text{Beispiel: } A = \begin{pmatrix} a_{11} & a_{12} & 0 & \cdots & 0 \\ a_{21} & a_{22} & a_{23} & \ddots & \vdots \\ 0 & a_{32} & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & a_{(n-1)n} \\ 0 & \cdots & 0 & a_{n(n-1)} & a_{nn} \end{pmatrix}$$

Tridiagonal-Systeme

Idee

Sei $A \in \mathbb{R}^{nxn}$ symmetrische und positiv definite Tridiagonalmatrix.
Man kann nun e_i $(1 \leq i < n)$ finden, sodass $\quad A = LDL^T$ mit:

$$L = \begin{pmatrix} 1 & 0 & \cdots & \cdots & 0 \\ e_1 & 1 & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & e_{n-1} & 1 \end{pmatrix}, \quad D = \begin{pmatrix} d_1 & 0 & \cdots & \cdots & 0 \\ 0 & d_2 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & d_n \end{pmatrix}$$

Pseudocode Tridiagonal-Systeme

Aus der Zerlegung folgt:

$$
\begin{aligned}
a_{11} &= d_1 & & \\
a_{k,k-1} &= e_{k-1} d_{k-1} & & k = 2{:}n \\
a_{kk} &= d_k + e_{k-1}^2 d_{k-1} = d_k + e_{k-1} a_{k,k-1} & & k = 2{:}n
\end{aligned}
$$

Es ergibt sich folgender Algorithmus:

$d_1 = a_{11}$
for $k = 2{:}n$
$\quad e_{k-1} = a_{k,k-1}/d_{k-1}$
$\quad d_k = a_{kk} - e_{k-1} a_{k,k-1}$
end

Pseudocode Tridiagonal-Systeme

Um nun $Ax = b$ zu lösen betrachten wir:
$Ly = b, \quad Dz = y$ und $\quad L^T x = z$

Es ergibt sich folgender Algorithmus:

for $k = 2{:}n$
$\quad t = e_{k-1}; e_{k-1} = t/d(k-1); d(k) = d(k) - te(k-1)$
end for $k = 2{:}n$
$\quad b(k) = b(k) - e(k-1)b(k-1)$
end
$b(n) = b(n)/d(n)$
for $k = n-1{:}-1{:}1$
$\quad b(k) = b(k)/d(k) - e(k)b(k+1)$
end

Der Algorithmus braucht $8n$ Flops.

Gliederung

Probleme

Probleme entstehen bei:

- Datenzugriff in langen Vektoren
- Überschreiben von Daten, die noch benötigt werden

Speichertechniken

Eine Lösung sind geeignete Speichertechniken, mit denen Schleifen innerhalb der Algorithmen eingespart werden.
u. a bieten sich folgende Methoden an:

- Compressed Sparse Row (CSR)
- Band-Einträge in einem Array speichern

Zusammenfassung

- Einsparung von Schleifen innerhalb der Algorithmen
- Reduktion des Speicherverbrauchs
- Algorithmen werden teilweise komplizierter

Mithilfe von auf Bandmatrizen angepassten Algorithmen lässt sich die Rechenzeit deutlich senken und somit lassen sich viele reale Probleme schneller berechnen.