Beispiel 176 (MA2 Sammlung)

LVA 118.153, Übungsrunde 12, 29.06.

Markus Nemetz, markus.nemetz@tuwien.ac.at, TU Wien, 06/2006

1 Angabe

Man löse das lineare Gleichungssystem

$$\begin{array}{rcrcrcr} -x_1 & + & 5x_2 & - & 2x_3 & = & 3 \\ x_1 & + & x_2 & - & 4x_3 & = & -9 \\ 4x_1 & - & x_2 & + & 2x_3 & = & 8 \end{array}$$

unter Anwendung des unter Anwendung des Einzelschrittverfahrens von Gauß-Seidel.

2 Theorie: Wichtige Vektor- und Matrixnormen

Vektornorm	**Name**	**zugehörige Matrixnorm**	**Name**
$\|x\|_1 = \sum_i \|x_i\|$	Betragssummennorm	$\|A\|_1 = max_k \sum_i \|a_{i,k}\|$	Spaltensummennorm
$\|x\|_2 = \sqrt{\sum_i \|x_i\|^2}$	Euklidische Norm	$\|A\|_F = \sqrt{\sum_{i,k} \|a_{i,k}\|^2}$	Frobenius-Norm
		$\|A\|_2 = \sqrt{max_i \|\lambda_i(A^T \cdot A)\|}$	Spektralnorm
$\|x\|_\infty = max_i \|x_i\|$	Maximumnorm	$\|A\|_\infty = max_i \sum_k \|a_{i,k}\|$	Zeilensummennorm

3 Theorie: Gauss-Seidel-Verfahren

Das Verfahren in Einzelschritten, auch Gauss-Seidel-Verfahren genannt, bedient sich wiederum der additiven Zerlegung der Koeffizientenmatrix in $A = L+D+R$ und hat folgende Iterationsvorschrift:

$$(L + D) \cdot x_{i+1} = -R \cdot x_i + a$$

Dieses Verfahren konvergiert, wenn die Koeffizentenmatrix diagonaldominant ist.

Die Iterationsvorschrift lautet konkreter:

$$x_k^{(m+1)} := \frac{1}{a_{k;k}} \left(b_k - \sum_{i=1}^{k-1} a_{k;i} \cdot x_i^{(m+1)} - \sum_{i=k+1}^{n} a_{k;i} \cdot x_i^{(m)} \right)$$

4 Lösung des Beispiels

4.1 Überprüfung auf Zeilensummennorm

$$\begin{array}{rcrcrcrll} -x_1 & + & 5x_2 & - & 2x_3 & = & 3 & i=1 & 5+2<1 \\ x_1 & + & x_2 & - & 4x_3 & = & -9 & i=2 & 1+4<1 \\ 4x_1 & - & x_2 & + & 2x_3 & = & 8 & i=3 & 4+1<2 \end{array}$$

Die Zeilensummennorm wird nicht erfüllt, daher formen die die Matrix so um, dass sie diagonaldominant wird (grosse Zahlen in der Hauptdiagonale - Vertausche Zeile 1 mit 3 und dann Zeile 2 mit 3):

$$\begin{array}{rcrcrcrll} 4x_1 & - & x_2 & + & 2x_3 & = & 8 & i=1 & 1+2<4 \\ -x_1 & + & 5x_2 & - & 2x_3 & = & 3 & i=2 & 1+2<5 \\ x_1 & + & x_2 & - & 4x_3 & = & -9 & i=3 & 1+1<4 \end{array}$$

4.2 Durchführung der Iteration

Anwendung der Rekursionsformel, Berechnung von $x_{k+1}^{[i]}$:

$$\begin{aligned} i=1: \quad & x_{k+1}^{[1]} = \frac{1}{a_{11}} \cdot (-\sum_{j<1} a_{1j} x_{k+1}^{[j]} - \sum_{j>1} a_{1j} x_k^{[j]} + b_1) = \\ & \frac{1}{4} \cdot (+1) \cdot x_k^{[2]} + \frac{1}{4} \cdot (-1) \cdot x_k^{[3]} + \frac{1}{4} \cdot 8 \\ i=2: \quad & x_{k+1}^{[2]} = \frac{1}{a_{22}} \cdot (-\sum_{j<2} a_{2j} x_{k+1}^{[j]} - \sum_{j>2} a_{2j} x_k^{[j]} + b_2) = \\ & \frac{1}{5} \cdot (1) \cdot x_k^{[1]} + \frac{1}{5} \cdot (2) \cdot x_k^{[3]} + \frac{1}{5} \cdot 3 \\ i=3: \quad & x_{k+1}^{[3]} = \frac{1}{a_{33}} \cdot (-\sum_{j<3} a_{3j} x_{k+1}^{[j]} - \sum_{j>3} a_{3j} x_k^{[j]} + b_3) = \\ & \frac{1}{-4} \cdot (-1) \cdot x_k^{[1]} + \frac{1}{-4} \cdot (-1) \cdot x_k^{[2]} + \frac{1}{-4} \cdot (-9) \end{aligned}$$

Erhalten mit dem Iterationsvektor $(x_0, y_0, z_0)^T = (0, 0, 0)^T$ ersten Iterationsdurchlauf:

$$x_1^{[1]} = \frac{1}{4}x_0^{[2]} - \frac{1}{2}x_0^{[3]} + 2 = 2$$
$$x_1^{[2]} = \frac{1}{5}x_1^{[1]} + \frac{2}{5}x_0^{[3]} + \frac{3}{5} = 1$$
$$x_1^{[3]} = \frac{1}{4}x_1^{[1]} + \frac{1}{4}x_1^{[2]} + \frac{9}{4} = 3$$

Erhalten mit dem Iterationsvektor $(x_1, y_1, z_1)^T = (2, 1, 3)^T$ ersten Iterationsdurchlauf:

$$x_2^{[1]} = \frac{1}{4}x_1^{[2]} - \frac{1}{2}x_1^{[3]} + 2 = 0.75$$
$$x_2^{[2]} = \frac{1}{5}x_2^{[1]} + \frac{2}{5}x_1^{[3]} + \frac{3}{5} = \frac{39}{20}$$
$$x_2^{[3]} = \frac{1}{4}x_2^{[1]} + \frac{1}{4}x_2^{[2]} + \frac{9}{4} = \frac{117}{14}$$

Nächster Iterationsdurchlauf mit Iterationsvektor $(x_2, y_2, z_2)^T = (0.75, \frac{39}{20}, \frac{117}{14})^T$.
Erhalte mit MATLAB nach 8 Iterationen Lösungen $x_1 = 1$, $x_2 = 2$, $x_3 = 3$ (siehe Anhang - mit Defektberechnung).

```
% GAUSS-SEIDEL-ITERATION
% BASED ON Preuss, Numerische Mathematik, p96
% SUITABLE ONLY FOR LINEAR EQUATIONS WITH 3x3 COEFF. MATRIX
% SEE ALSO Yang, 101

function  Y=gausseidel(A,B,X0,kmax)
%warnig set off (e.g. for rank deficiency warning)
warning off

%%Input Handling (general)
%too low number of input arguments
if nargin<4,
    sprintf('Usage gausseidel(A,B,X0,kmax) \n Remember A*x=b \n A \t
\t 3x3 matrix (A) \n B \t \t 3x1 matrix (b) \n x0 \t starting vector
\n kmax \t Maximum of Iterations')
    error(' ')
end

%no 3x3 coefficient matrix for A assigned
Asize=size(A);
if Asize(1)~=3 | Asize(2)~=3,
    sprintf('You must use a 3x3 coefficient matrix for A')
    error(' ')
end

%no 3x1 matrix for b assigned
Bsize=size(B);
if Bsize(1)~=3 | Bsize(2)~=1,
    sprintf('You must use a 1x3 matrix for B')
    error(' ')
end

%no 3x1 matrix for X0 assigned
Xsize=size(X0);
if Xsize(1)~=3 | Xsize(2)~=1,
    sprintf('You must use a 1x3 matrix for X0')
    error(' ')
end

%noninteger for kmax assigned
if kmax < 0 || ~isnumeric(kmax),
    kmax=100;
    sprintf('You used a noninteger for kmax - kmax set to 100')
end

%%warning for nondiagonaldominant coefficient matrix
diagA=diag(A);
if diagA(1)~=diagA(2) || diagA(2)~=diagA(3) || diagA(1)~=diagA(3),
    sprintf('Warning! Nondiagonaldominant coefficient matrix - maybe
no convergence!')
    tocontinue=input('Continue? Type Y or N >','s');
    if tocontinue=='Y'
        tobecontinued='yes'
    else
        error('You stopped the script')
    end
end

%some vars
NA=size(A,1); NB=size(B,2);
X =X0;

%first iteration (k=0)
```

```
ersteiter=A*X0\B;
if isnumeric(ersteiter),
    ersterg=[0; 0; 0];
else
    ersterg=ersteiter;
end
defekt=sqrt((max(eig((A*X-B)'*(A*X-B))))/3); %spectralnorm better
written
Y=sprintf('Schritt \t x \t \t \t y \t \t \t z \t \t Defekt \n0 \t \t
%f \t %f \t %f \t %e \n',ersterg(1),ersterg(2),ersterg(3),defekt);

%Iteration Loop
for k=1: kmax
    X(1,:) =(B(1,:)-A(1,2:NA)*X(2:NA,:))/A(1,1);
    for m=2:NA-1
        tmp =B(m,:)-A(m,1:m-1)*X(1:m-1,:)-A(m,m+1:NA)*X(m+1:NA,:);
        X(m,:) =tmp/A(m,m);
    end
    X(NA,:) =(B(NA,:)-A(NA,1:NA-1)*X(1:NA-1,:))/A(NA,NA);
    defekt=sqrt((max(eig((A*X-B)'*(A*X-B))))/3); %spectralnorm better
written
    td=sprintf('%d \t \t %f \t %f \t %f \t %e \n',k,X(1),X(2),X
(3),defekt);
    Y=[Y td];
    if defekt<1.0*10^-10,
        break;
    end
    X0=X;
end

% >> gausseidel(A,B,X0,15)
%
% Schritt    x           y           z       Defekt
% 0          0.000000    0.000000    0.000000    7.164728e+000
% 1          2.000000    1.000000    3.000000    4.509250e+000
% 2          0.750000    1.950000    2.925000    6.409628e-001
% 3          1.025000    1.975000    3.000000    1.127312e-001
% 4          0.993750    1.998750    2.998125    1.602407e-002
% 5          1.000625    1.999375    3.000000    2.818281e-003
% 6          0.999844    1.999969    2.999953    4.006018e-004
% 7          1.000016    1.999984    3.000000    7.045703e-005
% 8          0.999996    1.999999    2.999999    1.001504e-005
% 9          1.000000    2.000000    3.000000    1.761426e-006
% 10         1.000000    2.000000    3.000000    2.503761e-007
% 11         1.000000    2.000000    3.000000    4.403564e-008
% 12         1.000000    2.000000    3.000000    6.259403e-009
% 13         1.000000    2.000000    3.000000    1.100891e-009
% 14         1.000000    2.000000    3.000000    1.564852e-010
% 15         1.000000    2.000000    3.000000    2.752249e-011
```