Jacobi, Gauß-Seidel und SOR

Proseminar Numerische Mathematik

Alexander Oldemeier

8.7.2017

Problemstellung

- $Ax = b$, $A \in \mathbb{R}^{n \times n}$, $x \in \mathbb{R}^n$, $a \in \mathbb{R}^n$, $n \in \mathbb{N}$ soll gelöst werden.
- Iterative Verfahren vs direkte Verfahren.
- Kriterien: Geschwindigkeit, Einsatzmöglichkeiten, Stabilität.

Problemstellung

- $Ax = b$, $A \in \mathbb{R}^{n \times n}$, $x \in \mathbb{R}^n$, $a \in \mathbb{R}^n$, $n \in \mathbb{N}$ soll gelöst werden.
- Iterative Verfahren vs direkte Verfahren.
- Kriterien: Geschwindigkeit, Einsatzmöglichkeiten, Stabilität.

Splitting

- Wähle invertierbare Matrix B.
- $A = B + (A - B)$
- $Ax = b \iff x = \left(I_n - B^{-1}A\right)x + B^{-1}b$

Splitting

- Wähle invertierbare Matrix B.
- $A = B + (A - B)$
- $Ax = b \iff x = \left(I_n - B^{-1}A\right)x + B^{-1}b$

Splitting

- Wähle invertierbare Matrix B.
- $A = B + (A - B)$
- $Ax = b \iff x = \left(I_n - B^{-1}A\right) x + B^{-1}b$

Fixpunktgleichung

- Fixpunktgleichung: $x = \left(I_n - B^{-1}A\right)x + B^{-1}b$
- Fixpunktiteration: $x_{k+1} := Mx_k + c$ mit $M := \left(I_n - B^{-1}A\right)$, $c := B^{-1}b$.

Fixpunktgleichung

- Fixpunktgleichung: $x = \left(I_n - B^{-1}A\right) x + B^{-1}b$
- Fixpunktiteration: $x_{k+1} := Mx_k + c$ mit $M := \left(I_n - B^{-1}A\right)$, $c := B^{-1}b$.

Splitting-Verfahren

- Wir werden sehen: Unter gewissen Bedingungen konvergiert die Fixpunktiteration gegen die Lösung des LGS.
- Numerische Verfahren dieser Art nennt man Splitting-Verfahren.
- Iterative Verfahren vs. direkte Verfahren.

Splitting-Verfahren

- Wir werden sehen: Unter gewissen Bedingungen konvergiert die Fixpunktiteration gegen die Lösung des LGS.
- Numerische Verfahren dieser Art nennt man Splitting-Verfahren.
- Iterative Verfahren vs. direkte Verfahren.

Splitting-Verfahren

- Wir werden sehen: Unter gewissen Bedingungen konvergiert die Fixpunktiteration gegen die Lösung des LGS.
- Numerische Verfahren dieser Art nennt man Splitting-Verfahren.
- Iterative Verfahren vs. direkte Verfahren.

Der Banachsche Fixpunktsatz

Satz

Sei $(\mathcal{K}, \| \, . \, \|)$ *ein vollständiger metrischer Raum und* $\Phi : \mathcal{K} \to \mathcal{K}$ *eine Kontraktion, d.h. eine Abbildung, die für ein* $q < 1, q \in \mathbb{R}$ *die Eigenschaft*

$$\| \, \Phi(x) - \Phi(z) \, \| \leq q \, \| \, x - z \, \| \; \forall x, z \in \mathcal{K}$$

erfüllt. Dann hat die Fixpunktgleichung $x = \Phi(x)$ *genau eine Lösung* $\hat{x} \in \mathcal{K}$ *und für die Fixpunktiteration* $x_{k+1} := \Phi(x_k)$ *gilt* $\lim\limits_{k \to \infty} x_k = \hat{x}$.

Konvergenz und Matrixnorm

Lemma

Sei $|||.|||$ eine Matrixnorm in $\mathbb{R}^{n\times n}$, die mit einer Vektornorm $\|.\|$ verträglich ist. Die Fixpunktiteration konvergiert gegen $\hat{x} = A^{-1}b$, wenn $|||M||| < 1$.

Konvergenz und Spektralradius

Satz

Das Fixpunktverfahren konvergiert genau dann wenn $\rho(M) < 1$.

Konvergenzgeschwindigkeit

Satz

Für das Fixpunktverfahren mit Iterationsmatrix M, $\rho(M) < 1$ und Lösung $\hat{x}$ gilt:

$$\rho(M) = \alpha := \sup_{x_0} \limsup_{k \to \infty} \left(\|x^k - \hat{x}\| \right)^{\frac{1}{k}}$$

Das Jacobi-Verfahren

- Split:

$$A = \underbrace{\begin{pmatrix} a_{11} & 0 & \cdots & 0 \\ & \ddots & & \vdots \\ 0 & & 0 & \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & \cdots & a_{nn} \end{pmatrix}}_{:=D} - \underbrace{\begin{pmatrix} 0 & 0 & \cdots & 0 \\ -a_{21} & \ddots & 0 & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ -a_{n1} & -a_{n2} & \cdots & 0 \end{pmatrix}}_{:=L} - \underbrace{\begin{pmatrix} -a_{11} & -a_{12} & \cdots & -a_{1n} \\ & \ddots & & \\ 0 & & \ddots & -a_{2n} \\ \vdots & & & \vdots \\ 0 & \cdots & \cdots & 0 \end{pmatrix}}_{:=R}$$

- Jacobi-Verfahren: $B = D$, $x_{k+1} := \underbrace{D^{-1}(L+R)}_{:=M_J} x_k + \underbrace{D^{-1}b}_{:=c_J}$

- Herleitung über komponentenweise Darstellung: $x_i = \dfrac{b_i - \sum_{\substack{j=1 \\ j \neq i}}^{n} a_{ij} x_j}{a_{ii}}$

Das Jacobi-Verfahren

- Split:

$$A = \underbrace{\begin{pmatrix} a_{11} & 0 & \cdots & 0 \\ & \ddots & & \vdots \\ 0 & & 0 & \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & \cdots & a_{nn} \end{pmatrix}}_{:=D} - \underbrace{\begin{pmatrix} 0 & 0 & \cdots & 0 \\ -a_{21} & \ddots & 0 & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ -a_{n1} & -a_{n2} & \cdots & 0 \end{pmatrix}}_{:=L} - \underbrace{\begin{pmatrix} -a_{11} & -a_{12} & \cdots & -a_{1n} \\ & \ddots & & \\ 0 & & \ddots & -a_{2n} \\ \vdots & & & \vdots \\ 0 & \cdots & \cdots & 0 \end{pmatrix}}_{:=R}$$

- Jacobi-Verfahren: $B = D$, $x_{k+1} := \underbrace{D^{-1}(L+R)}_{:=M_J} x_k + \underbrace{D^{-1}b}_{:=c_J}$

- Herleitung über komponentenweise Darstellung: $x_i = \dfrac{b_i - \sum_{\substack{j=1 \\ j \neq i}}^{n} a_{ij} x_j}{a_{ii}}$

Das Jacobi-Verfahren

- Split:

$$A = \underbrace{\begin{pmatrix} a_{11} & 0 & \cdots & 0 \\ 0 & \ddots & 0 & \vdots \\ \vdots & 0 & \ddots & \vdots \\ 0 & \cdots & \cdots & a_{nn} \end{pmatrix}}_{:=D} - \underbrace{\begin{pmatrix} 0 & 0 & \cdots & 0 \\ -a_{21} & \ddots & 0 & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ -a_{n1} & -a_{n2} & \cdots & 0 \end{pmatrix}}_{:=L} - \underbrace{\begin{pmatrix} -a_{11} & -a_{12} & \cdots & -a_{1n} \\ 0 & \ddots & \ddots & -a_{2n} \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & 0 \end{pmatrix}}_{:=R}$$

- Jacobi-Verfahren: $B = D$, $x_{k+1} := \underbrace{D^{-1}(L+R)}_{:=M_J} x_k + \underbrace{D^{-1}b}_{:=c_J}$

- Herleitung über komponentenweise Darstellung: $x_i = \dfrac{b_i - \sum\limits_{\substack{j=1 \\ j \neq i}}^{n} a_{ij} x_j}{a_{ii}}$

Das Jacobi-Verfahren

- Herleitung über komponentenweise Darstellung: $x_i = \frac{b_i - \sum_{\substack{j=1 \\ j \neq i}}^{n} a_{ij} x_j}{a_{ii}}$
- Das Verfahren ist vollständig parallelisierbar (Gesamtschrittverfahren).

Das Jacobi-Verfahren

- Herleitung über komponentenweise Darstellung: $x_i = \dfrac{b_i - \sum\limits_{\substack{j=1 \\ j \neq i}}^{n} a_{ij} x_j}{a_{ii}}$
- Das Verfahren ist vollständig parallelisierbar (Gesamtschrittverfahren).

Das Jacobi-Verfahren

Konvergenz für diagonaldominante Matrizen

Satz

Das Jacobi-Verfahren konvergiert für alle strikt diagonaldominanten Matrizen $A \in \mathbb{R}^{n \times n}$, d.h. wenn $|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}| \; \forall i \in \{1, ..., n\}$.

Das Gauß-Seidel-Verfahren

- Idee: Konvergenzbeschleunigung durch Verwendung der neuen Werte:

$$x_i^{(k+1)} = \frac{b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)}}{a_{ii}}$$

- Gauß-Seidel: $B = D - L$, $x_{k+1} = \underbrace{(D-L)^{-1} R}_{:=M_{GS}} x_k + \underbrace{(D-L)^{-1} b}_{:=c_{GS}}$

Das Gauß-Seidel-Verfahren

- Idee: Konvergenzbeschleunigung durch Verwendung der neuen Werte:

$$x_i^{(k+1)} = \frac{b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)}}{a_{ii}}$$

- Gauß-Seidel: $B = D - L$, $x_{k+1} = \underbrace{(D-L)^{-1} R}_{:=M_{GS}} x_k + \underbrace{(D-L)^{-1} b}_{:=c_{GS}}$

Das Gauß-Seidel-Verfahren

Konvergenz für diagonaldominante Matrizen

Satz

Das Gauß-Seidel-Verfahren konvergiert für alle strikt diagonaldominanten Matrizen.

- Wenn das Verfahren konvergiert, konvergiert es schneller als das Jacobi-Verfahren ($\rho(M_{GS}) = \rho(M_J)^2$).

Das Gauß-Seidel-Verfahren

Konvergenz für diagonaldominante Matrizen

Satz

Das Gauß-Seidel-Verfahren konvergiert für alle strikt diagonaldominanten Matrizen.

- Wenn das Verfahren konvergiert, konvergiert es schneller als das Jacobi-Verfahren ($\rho(M_{GS}) = \rho(M_J)^2$).

Konvergenzbeschleunigung

Successive-Over-Relaxation (SOR)

- Idee: Konvergenzbeschleunigung durch Gewichtung der neuen Werte:

$$x_i^{(k+1)} := x_i^{(k)} + \omega \left(\frac{b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)}}{a_{ii}} - x_i^{(k)} \right)$$

- SOR: $B = \frac{1}{\omega} D - L$,

$$x_{k+1} = \underbrace{(D - \omega L)^{-1} [(1 - \omega) D + \omega R]}_{:= M_{SOR}} x_k + \underbrace{\omega (D - L)^{-1} b}_{:= c_{SOR}}$$

Konvergenzbeschleunigung

Successive-Over-Relaxation (SOR)

- Idee: Konvergenzbeschleunigung durch Gewichtung der neuen Werte:

$$x_i^{(k+1)} := x_i^{(k)} + \omega \left(\frac{b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)}}{a_{ii}} - x_i^{(k)} \right)$$

- SOR: $B = \frac{1}{\omega} D - L$,

$$x_{k+1} = \underbrace{(D - \omega L)^{-1} \left[(1 - \omega) D + \omega R \right]}_{:= M_{SOR}} x_k + \underbrace{\omega (D - L)^{-1} b}_{:= c_{SOR}}$$

Konvergenzbeschleunigung

Der Satz von Ostrowski und Reich

Satz

Das SOR-Verfahren konvergiert für positiv definitive symmetrische Matrizen genau dann wenn $\omega \in]0, 2[$.

- Problem: Finde optimales ω.
- Zur Lösung des Problems muss man die Eigenwerte von M_{SOR} kennen.

Konvergenzbeschleunigung

Der Satz von Ostrowski und Reich

Satz

Das SOR-Verfahren konvergiert für positiv definitive symmetrische Matrizen genau dann wenn $\omega \in]0, 2[$.

- Problem: Finde optimales ω.
- Zur Lösung des Problems muss man die Eigenwerte von M_{SOR} kennen.

Konvergenzbeschleunigung

Der Satz von Ostrowski und Reich

Satz

Das SOR-Verfahren konvergiert für positiv definitive symmetrische Matrizen genau dann wenn $\omega \in]0, 2[$.

- Problem: Finde optimales ω.
- Zur Lösung des Problems muss man die Eigenwerte von M_{SOR} kennen.

Konvergenzbeschleunigung

Tschebyscheff-Beschleunigung

- Idee: Konvergenzbeschleunigung durch Linearkombination der bisherigen Iterationen $y_k = \sum_{i=0}^{k} \alpha_{ki} x_k$
- Minimiere Fehler $y_k - \hat{x} := d_k$
- Stelle d_k dar als $P_k(M)\, d_0$ mit $P_K(M)$ Matrixpolynom.
- Minimiere $|||P_k(M)|||_2$ mit Hilfe der Theorie der Tschebyscheff-Polynome.

Konvergenzbeschleunigung

Tschebyscheff-Beschleunigung

- Idee: Konvergenzbeschleunigung durch Linearkombination der bisherigen Iterationen $y_k = \sum_{i=0}^{k} \alpha_{ki} x_k$
- Minimiere Fehler $y_k - \hat{x} := d_k$
- Stelle d_k dar als $P_k(M)\, d_0$ mit $P_K(M)$ Matrixpolynom.
- Minimiere $|||P_k(M)|||_2$ mit Hilfe der Theorie der Tschebyscheff-Polynome.

Konvergenzbeschleunigung

Tschebyscheff-Beschleunigung

- Idee: Konvergenzbeschleunigung durch Linearkombination der bisherigen Iterationen $y_k = \sum_{i=0}^{k} \alpha_{ki} x_k$
- Minimiere Fehler $y_k - \hat{x} := d_k$
- Stelle d_k dar als $P_k(M)\, d_0$ mit $P_K(M)$ Matrixpolynom.
- Minimiere $|||P_k(M)|||_2$ mit Hilfe der Theorie der Tschebyscheff-Polynome.

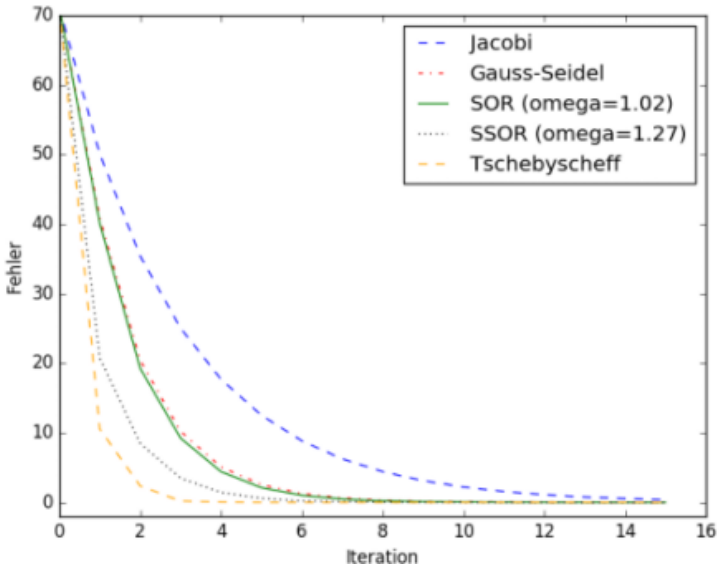
Konvergenzbeschleunigung

Tschebyscheff-Beschleunigung

- Idee: Konvergenzbeschleunigung durch Linearkombination der bisherigen Iterationen $y_k = \sum_{i=0}^{k} \alpha_{ki} x_k$
- Minimiere Fehler $y_k - \hat{x} := d_k$
- Stelle d_k dar als $P_k(M)\, d_0$ mit $P_K(M)$ Matrixpolynom.
- Minimiere $|||P_k(M)|||_2$ mit Hilfe der Theorie der Tschebyscheff-Polynome.

Implementierung und Vergleich

$$\begin{pmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{pmatrix} x = \begin{pmatrix} 19 \\ 45 \\ 0 \end{pmatrix}$$



Approximationsfehler

	Jacobi	GS	SOR(1.02)	SSOR(1.27)	Tschebyscheff
0	71.1573959613	71.1573959613	71.1573959613	71.1573959613	71.1573959613
1	50.091166886	40.875	40.1234173916	20.7480263934	10.5658983625
2	35.4198037826	20.4375	19.2730510986	8.49053949052	2.37506100576
3	25.045583443	10.21875	9.23819526619	3.4925737569	0.199798972825
4	17.7099018913	5.109375	4.42848563671	1.43932086628	0.0412576349189
5	12.5227917215	2.5546875	2.12286325483	0.593848341951	0.00325115046078
6	8.85495094566	1.27734375	1.01762754611	0.245198554277	0.000748112360541
7	6.26139586075	0.638671875	0.487815602233	0.101290942402	5.6822856305e-05
8	4.42747547283	0.3193359375	0.233842001176	0.0418563672888	1.25540186679e-05
9	3.13069793037	0.15966796875	0.112095802724	0.0172999420843	1.04542906106e-06
10	2.21373773641	0.079833984375	0.0537348676675	0.0071513865992	2.17127675838e-07
11	1.56534896519	0.0399169921875	0.0257586451329	0.00295650703879	1.7189934335e-08
12	1.10686886821	0.0199584960937	0.0123478074457	0.00122235602233	3.92121418636e-09
13	0.782674482594	0.00997924804687	0.0059191136774	0.000505403135294	3.00650520098e-10
14	0.553434434104	0.00498962402344	0.00283741926491	0.000208974634498	6.64637576901e-11
15	0.391337241297	0.00249481201172	0.00136016108555	8.64092957337e-05	5.43316794448e-12