Zweidimensionales sor-Verfahren

erstellt von

Kittel Matthias

Im Rahmen des

Mathematischen Praktikums

Sommersemester 2000
3/30/00

1. Einleitung

In diesem Praktikumsbeispiel sollte ein Iterationsverfahren an einem konkreten Beispiel implementiert werden. Als solches Verfahren wurde das sor (succesive overrealxing)-Verfahren ausgewählt und angepasst. Ein Programmcode wurde erstellt und im Anhang sind die Ergebnisse als Plots und Datenfiles beigefügt.

2. Mathematische Grundlagen

2.1. Allgemeines

Bei stationären iterativen Verfahren zur Lösung linearer Gleichungssysteme wird ausgehend von einem Startwert

$x^{(0)}$
eine unendliche Folge

$x^{(k+1)} := T(x^{(k)})$
$k = 0,1,2,3,...$
definiert.

Um auf Grundlage einer Iterationsvorschrift T einen Algorithmus zur Lösung linearer Gleichungssysteme zu definieren, muss man obige Folge nach endlich vielen Schritten abbrechen. Dann tritt stets der Verfahrensfehler

$x^{(stop)} - x^*$
auf, wobei

x^*
der gesuchte Lösungsvektor ist.

Man muss nun, eine dem Problem angepasste Entscheidung treffen, wann das Verfahren abgebrochen werden soll. Das Abbruchkriterium sollte

.) die Iteration stoppen, wenn die gewünschte Genauigkeit erreicht wurde
.) abbrechen, wenn der Fehler nicht oder zu langsam kleiner wird
.) den maximalen Rechenaufwand für die Iteration limitieren.

2.2. Stationäre iterative Verfahren

Die Grundform stationärer Iterationsverfahren zur Lösung linearer Gleichungsysteme ist von der Bauart

$x^{(k)} := Bx^{(k-1)} + c,$

$k = 1,2,3,...$

$B \in R^{(n \times n)}, c \in R^n$

B und c von k unabhängig. Um ein lineares Gleichungssystem Ax = b in eine iterative Form zu bringen, kann man z.B. die Koeffizientenmatrix additiv in

A = L + D + U

Aufspalten. Die Summanden sind D = diag(a(1,1),…,a(n,n)), die strikte Dreiecksmatrix L und die rechte obere Dreiecksmatrix U.

Das Ausgangssystem lässt sich mit der Zerlegung als

Dx = b -Lx – Ux

Schreiben. Wenn A nicht singulär ist, lässt sich, eventuell nach Pivotierung, stets erreichen, dass D nichtsingulär ist, also alle a(i,i) ungleich Null sind:

$x = D^{-1}b - D^{-1}(L+U)x$

Mit

$B := -D^{-1}(L+U), c := D^{-1}b$

Erhält man daraus die Fixpunktform

x = T(x) := Bx + c

Zu den stationären iterativen Lösungsmethoden linearer Gleichungssysteme zählen in erster Linie folgende Verfahren:

.) *Jacobi-Verfahren*:
Ein Iterationsschritt des Jacobi- oder Gesamtschrittverfahrens entspricht der lokalen Lösung für eine Variable. Die Methode ist leicht implementierbar, konvergiert aber oft nur sehr langsam.
.) *Gauss-Seidel-Verfahren*:
Im Unterschied zum Jacobi-Verfahren werden beim Gauss-Seidel- oder Einzelschrittverfahren die erhaltenen Näherungswerte sofort nach ihrer Berechnung weiterverwendet. Die Konvergenzgeschwindigkeit erhöht sich dadurch aber meist immer noch verhältnismässig gering.
.) *sor-Verfahren*
werden aus dem Gauss-Seidel-Verfahren unter Einfuehrung eines Extrapolationsparameters omega abgeleitet. Bei (fast) optimaler Wahl von omega kann eine wesentliche Konvergenzbeschleunigung erreicht werden.

2.3. Gauss-Seidel-Verfahren

Die einzelnen Gleichungen des Systems Ax = b kann man auch geordnet auflösen, wobei man bereits berechnete Werte sofort im nächsten Rechenschritt verwendet:

do i=1,2,3,…
  do j=1,2,3,..,n

$$x_j^{(k)} := (b_i \sum_{k=1}^{j-1} a_{jk} x_k^{(k)} - \sum_{k=j+1}^{n} a_{jk} x_k^{(k-1)}) / a_{jj}$$

  enddo
  if Abbruchkriterium erfüllt **then exit**
enddo

In diesem Fall müssen die Berechnungen seriell ausgeführt werden, da jede Komponente der neuen Iteration von allen zuvor berechneten Komponenten abhängt.

In Matrixschreibweise, mit oben eingeführten Abkürzungen stellt sich das Gauss-Seidel-Verfahren wie folgt da:

$$x^{(k)} := (D+L)^{-1}(b - Ux^{(k-1)}) = x^{(k-1)} - (D+L)^{-1}(Ax^{(k-1)} - b),$$
$$k = 1,2,3,...$$

Für eine reguläre, strikt diagonaldominante Matrix, konvergiert das Gauss-Seidel-Verfahren für jeden Startvektor. Die Konvergenzgeschwindigkeit ist zumindest gleich gross oder grösser wie beim Jacobi-Verfahren.
Konvergenz kann auch, für in der Regl häufig auftretenden symmetrischen, positiv definiten Matrizen, sichergestellt werden.

2.4. sor-Verfahren

Durch komponentenweise Extrapolation in Form eines gewichteten Durchschnitts zwischen dem Resultatvektor der vorhergehenden Iteration und dem neuen Gauss-Seidel-Wert erhält man die succsive overrelaxation, das sor-Verfahren

do k=1,2,3,…
  do I=1,2,…,n

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

  enddo
  if Abbruchkriterium erfüllt **then exit**
enddo

Da meist omega > 1 gewählt wird, spricht amn von Überrelaxation. Dieser Überrelaxationsfaktor omega soll die Konvergenz möglichst stark beschleunigen. Im allgemeinen Fall ist es unmöglich, günstige Werte von omega im voraus zu ermitteln. Alle sor-implementierungen enthalten Schätzmechanismen für einen möglichst vorteilhaften Wert von omega.

Eine Einschränkung für die Wahl von omega ergibt sich aus dem **Satz** von *Kahan*:

DerSpektralradius der Iterationsmatrix B der sor-Verfahrens genügt der Ungleichung

$$\rho(B) \geq |\omega - 1|, \forall \omega \in R$$

Satz von *Ostrowski* und *Reich*:

Ist A ein reelle n-mal-n Matrix, symmetrisch und positiv definit, dann konvergiert das sor-Verfahren für jedes omega zwischen 0 und 2 und jedem Stratwert x(0).

Die Konvergenz ist damit für alle omega zwischen 0 und 2 gegeben, wobei die Geschwindigkeit sehr unterschiedlich sein kann.

3. Ein konkretes Beispiel

Als Beispiel für die Implementierung des sor-Verfahrens habe ich folgendes Problem gewählt:
Ein Randwertproblem der Laplace-Gleichung auf einem rechteckigen Gebiet.
Eine dünne rechteckige Stahlplatte, an deren Rändern sowohl konstante Temperaturen oder mögliche Temperaturableitung am Rand vorgegeben werden können.

Suche u(x,y), sodass gilt:

$$\partial^2 u / \partial x^2 + \partial^2 u / \partial y^2 = 0$$

Der Stempel für dieses Problem sieht folgendermassen aus:

$$\nabla^2 u_{ij} = 1/h^2 \begin{Bmatrix} & 1 & \\ 1 & -4 & 1 \\ & 1 & \end{Bmatrix} u_{ij} = 0$$

Die Iterationsvorschrift im sor-Verfahren lautet:

$$u_{ij}^{(k+1)} = u_{ij}^{(k)} + \omega\left[(u_{i+1,j}^{(k)} + u_{i-1,j}^{(k)} + u_{i,j+1}^{(k)} + u_{i,j-1}^{(k)} - 4u_{ij}^{(k)})/4\right]$$

Weiters ist das Programm auch auf folgenden Fall anwendbar:

$$\nabla^2 u = -Q/k = 1/h^2 \begin{Bmatrix} & 1 & \\ 1 & -4 & 1 \\ & 1 & \end{Bmatrix} u_{ij}$$

Q ist die zu- oder abgeführte Wärmemenge in cal/cm**3, k für Stahl 0.16 cal/sec*cm**2*°C/cm.

Die Randbedingungen können als Dirichlet oder Neuman gesetzt werden.

4. Der Code

Der Code ist in FORTRAN 77 geschrieben und kommentiert im Anhang beigefügt.

Er bsteht aus folgenden Teilen:

.) sor-Makefile:
File, das das oftmalige compailieren und linken vereinfacht und beschleunigt

.) FORTRAN-includefile SORCOM:
File, in dem alle globalen Variablen und Parameter definiert beziehungsweise gesetzt werden, erleichter das filehandling.

.) FORTRAN-Programm MAIN:
Das Hauptprogramm, in dem das sor-Verfahren implementiert wurde.

.) FORTRAN-Subroutine IN:
Hier werden alle notwendigen sor-Parameter aus dem Parameterfile eingelesen und eventuelle Warnings ausgegeben.

.) FORTRAN-Subroutine LAPLACE:
In diesem File wird speziell für das oben beschrieben Beispiel die Einträge der Matrix gesetzt.

.) FORTRAN-Subroutine RES:
In diesem File wird die “rechte Seite” der Matrix definiert.

.) FORTRAN-Subroutine START:
Hier wird der Startvektor der Iteration eingegeben.

.) FORTRAN-Subroutine OUT:
Das Ausgabefile.

.) shell-script GO:
Diese Skript ist zum filehandling vorgesehen, es startet das sor-Programm sor.xl, linkt es mit dem Parameterfile und “verstaut” die Ergebnisfiles in den richtigen Verzeichnissen.

5. Rechenbeispiele

Erstes Beispiel:

Rechteckige Stahlplatte, 10 mal 20 Zentimeter, linker, oberer und unterer Rand auf 0°C, rechter Rand auf 100°C gehalten. Schrittweite 5 cm, das entspricht 3 inneren Punkten.
Hier die analytische Lösung:

u(1)=1.785714285714 °C
u(2)=7.142857142857 °C
u(3)= 26.78571428571°C

Bei einer Genauigkeit von 10e-3 und einm Startwert von 1 wurde der sor-Parameter omega variert:

Omega	Iterationen
1.00	6
1.02	5
1.05	5
1.0625	5
1.075	5
1.10	6
1.20	8
1.30	10
1.40	11
1.50	12

Jetzt wurde die Schrittweite auf 2.5 cm gesetzt und wieder der Parameter variiert, innere Punktanzahl N=21:

Omega	Iterationen
1.00	47
1.10	37
1.20	27
1.25	21
1.28	17
1.30	18
1.35	19
1.40	22
1.50	28
1.60	38
1.70	55
1.80	86
1.90	169

Setzen der Schrittweite auf 1.25 cm (N=105):

Omega	Iterationen
1.00	50
1.10	42
1.20	35
1.30	28
1.40	22
1.50	16
1.55	10
1.58	8
1.60	14
1.65	15
1.70	17

Schrittweite 1 cm (N=171):

Omega	Iterationen
1.00	245
1.10	202
1.20	165
1.30	132
1.40	103
1.50	76
1.55	62
1.60	42
1.62	36
1.65	46
1.70	57
1.80	88
1.90	158

Und schliesslich Schrittweite 0.5 cm (N=741):

Omega	Iterationen
1.00	809
1.10	671
1.20	553
1.30	452
1.40	363
1.50	283
1.60	211
1.70	141
1.75	103

1.78	72
1.80	76
1.82	93
1.85	97
1.90	152

Diese Berechnungen führen nun zu folgendem Schluss:

Gitterpunkte	Iterationen (Gauss)	Iterationen (sor)	Iter (Gauss)/Iter (sor)
3	6	5	1.20
21	19	8	2.38
105	50	8	5.56
171	65	11	5.92
741	128	18	7.11
1176	147	7	21.00

Je mehr Gitterpunkte, desto näher ist omega bei zwei, je mehr Gitterpunkte, desto effektiver ist das sor-Verfahren im Vergleich mit dem Gauss-Seidel-Verfahren

6. Anhang

Im Anhang befinden sich folgende Plots:

.) example 1
10 cm mal 20 cm Platte
Gitterpunkte 3
Linker Rand: 0°C
Rechter Rand: 100°C
Unterer Rand: 0°C
Oberer Rand: 0°C

.) example 2
10 cm mal 20 cm Platte
Gitterpunkte 21
Linker Rand: 0°C
Rechter Rand: 100°C
Unterer Rand: 0°C
Oberer Rand: 0°C

.) example 4
15 cm mal 20 cm Platte
Gitterpunkte 10
Linker Rand: Einfluss 1 °C
Rechter Rand: Einfluss 1°C
Unterer Rand: 0°C
Oberer Rand: 0°C

.) example 5
15 cm mal 20 cm Platte

Gitterpunkte 12
Linker Rand: 0°C
Rechter Rand: 0°C
Unterer Rand: Einfluss 1°C
Oberer Rand: Einfluss 1°C

.) example 6
10 cm mal 20 cm Platte
Gitterpunkte 6
Linker Rand: 20°C
Rechter Rand: 20°C
Unterer Rand: 20°C
Oberer Rand: Einfluss 1°C

.) example 7
10 cm mal 20 cm Platte
Gitterpunkte 3
Linker Rand: Einfluss 1°C
Rechter Rand: 0°C
Unterer Rand: 0°C
Oberer Rand: 0°C

.) relation – grid points/sor-parameter omege
mit Vergleichskurve log(x)/4

.) drei weiter Relationen von omega mit Gauss-Seidel, sor Iterationen und deren Verhältnis

.) der Code und alle zugehörigen Programme

.) Parameter-file und Ausgebefile für Beispiel 1

Quellenverzeichnis

[1] Christoph Überhuber, **Computernumerik 2**, Springer Verlag

[2] M. Firneis, **Rechenmethoden der Astronomie**, Vorlesung SS98, Universität Wien, Institut für Astronomie