universitätbonn

Institut für Numerische Simulation der Universität Bonn
Prof. Dr. Mario Bebendorf

Praktikum im Wintersemester 2013/2014

Programmierpraktikum numerische Algorithmen (P2E1)
(Numerische Lösung der Wärmeleitungsgleichung)

Betreuer: Christian Kuske[1]

Blatt 5

1 Einführung

Auf Blatt 4 wurde die Zweigitter-Methode eingeführt, die auf diesem Blatt zur Mehrgitter-Methode erweitert werden soll. Bei dem Verfahren mit zwei Gittern konnte durch wenige Iterationsschritte eine Näherung der Lösung bestimmt werden. Hierbei traten zwei wesentliche Probleme auf. Erstens muss auf dem groben Gitter die Lösung exakt bestimmt werden und zweitens benötigt das Verfahren bei einer ungünstigen Wahl des Startvektors zu viele Iterationsschritte.

Mehrere Gitter

Das erste Problem kann dadurch gelöst werden, dass auf dem groben Gitter das gleiche Verfahren rekursiv angewendet wird, da es sich vom ursprünglichen Problem lediglich durch die doppelte Gitterweite unterscheidet. Das kann wiederholt werden, bis lediglich ein innerer Punkt existiert, wodurch die Lösung direkt berechnet werden kann. Andernfalls kann durch die stark reduzierte Problemgröße die exakte Lösung schneller bestimmt werden als in der Zweigitter-Methode. Dieses Verfahren heißt Mehrgitter-Methode.

Startwert

Da bei einem allgemein zu lösenden Problem keine Kenntnis über dessen Verhalten vorliegt, muss ein brauchbarer Startvektor auf andere Weise ermittelt werden. Zur Ermittlung des Startvektors werden, beginnend bei dem gröbsten Gitter, einige Mehrgitterschritte durchgeführt, um eine Annäherung zu erhalten. Anschließend wird mit Hilfe dieses Startwerts die Mehrgitter-Methode erneut aufgerufen und führt zur Lösung des ursprünglichen Problems. Dieses Verfahren heißt vollständige Mehrgitter-Methode.

[1]We6 4.002; Tel.: 0228/73-60454; Email: kuske@ins.uni-bonn.de

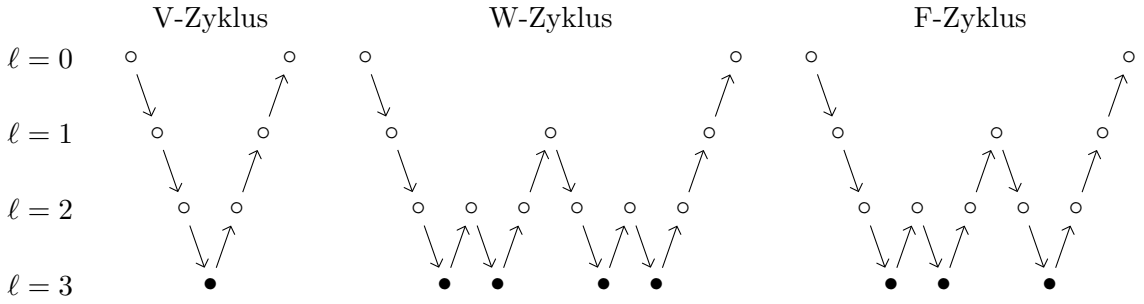

Abbildung 1: Darstellung einer Mehrgitter-Iteration mit Hilfe des V-, W-, und F-Zyklus für vier Level. Dabei sind die Glättungsschritte durch „$\circ$“ und die Lösungsschritte durch „$\bullet$“ gekennzeichnet.

2 Mehrgitter-Methode

In der Praxis werden in den meisten Fällen drei Arten von Mehrgitter-Methoden verwendet. Dazu zählen der V-, W- und F-Zyklus. Die ersten beiden unterscheiden sich durch die Anzahl der rekursiven Aufrufe auf einem Level. Im Gegensatz zum F-Zyklus. Dieser bestimmt zunächst den Startvektor mittels einiger V-Zyklen und führt danach einen V-Zyklus auf dem feinsten Gitter durch. Eine schematische Darstellung alle drei Zyklen ist in Abbildung 1 zu finden.

V- und W-Zyklus

Für diese Zyklen ist nachfolgende Algorithmus gegeben. Dabei entsteht für $\mu = 1$ der V-Zyklus und für $\mu = 2$ der W-Zyklus. Die Herkunft dieser Namen ist durch den rekursiven Aufruf begründet, da das μ-Schema ein V beziehungsweise ein W beschreibt.

```
    Glätte L_h u_h = f_h auf Ω_h
    Falls Ω_h das gröbste Gitter ist
      Gehe zu 11
    Sonst
      Berechne das Residuum r_h = f_h − L_h u_h
      Bestimme r_H = I_h^H r_h
      Setze e_H = 0
      Berechne L_H e_H = r_H durch μ Mehrgitterschritte
    Bestimme e_h = I_H^h e_H
    Setze u_h = u_h + e_h
11: Glätte L_h u_h = f_h auf Ω_h
```

Algorithmus 2.1: Mehrgitter-Methode (μ-Schema)

Die Durchführung des Algorithmus ist ähnlich zur Zweigitter-Methode mit zusätzlichen rekursiven Aufrufen. Außerdem wird die Lösung auf dem gröbsten Gitter ebenfalls durch den Glätter berechnet, so dass ein zusätzliches Lösen entfällt.

F-Zyklus

Der F-Zyklus ist vor allem bei Problemen anzuwenden, bei denen nicht auf einfache Weise ein guter Startvektor gewählt werden kann oder nicht gewählt werden soll. Diese Methode wird als vollständige Mehrgitter-Methode (full multigrid method) bezeichnet und lässt sich wie folgt implementieren.

```
   Falls Ω_h das gröbste Gitter ist
     Setze u_h = 0
     Gehe zu 08
   Sonst
     Bestimme f_H = I_h^H f_h
     Berechne u_H durch einen vollständigen Mehrgitterschritt
   Bestimme u_h = I_H^h u_H
08: Berechne L_h u_h = f_h durch einen V-Zyklus
```

Algorithmus 2.2: vollständige Mehrgitter-Methode

Ist die Lösung nach einem F-Zyklus nicht ausreichend genau, können weitere V- oder W-Zyklen nachgeschoben werden.

3 Aufgaben

a) Implementiere die Mehrgitter-Methode zur Lösung von $L_h u_h = f_h$ mit Hilfe der Funktion:

- `unsigned int MGM(const unsigned int n, double * const u, const double * const f, const double eps, const unsigned int depth, const unsigned int mu)`,

wobei der Rückgabewert der Anzahl der benötigten Iterationen entspricht. Setze $n = 2^L + 1$ und beende das Verfahren, falls $\|r_h\|_2 < \varepsilon$ gilt! Die Tiefe `depth` soll zwischen 3 und 5 gewählt werden. Der Parameter μ spiegelt die Anzahl der rekursiven Aufrufe von `MGM` auf dem gleichen Level wieder und ist gleich 1 für den V-Zyklus und gleich 2 für den W-Zyklus. Verwende als Glätter und Löser das SOR-Verfahren mit `maxIt` $= 5$!

b) Implementiere die vollständige Mehrgitter-Methode (F-Zyklus) zur Bestimmung eines Startvektors u_h für $L_h u_h = f_h$ mit Hilfe der Funktion:

- `void FMGM(const unsigned int n, double * const u, const double * const f, const unsigned int depth)`,

wobei $n = 2^L + 1$ sei und die Mehrgitter-Methode mit einer Genauigkeit von $\varepsilon = 10^{-4}$ und $\mu = 1$ verwendet werden soll.

c) Erstelle eine Tabelle für verschiedene n und vergleiche die Anzahl der Iterationsschritte und die Rechenzeit von V- und W-Zyklen!

d) Erstelle eine weitere Tabelle wie in Aufgabe c), wobei zusätzlich ein F-Zyklus vor den V- und W-Zyklen ausgeführt wird!