Universität Hamburg
Fachbereich Informatik
Wissenschaftliches Rechnen
Prof. T. Ludwig, H. Lenhart, U. Körner, N. Hübbe

Übungsblatt 4 zum Praktikum
Paralleles Programmieren für
Geowissenschaftler im SS 2012
Abgabe: 22.05.2012

Aufgabe 4: Lösung der Poissongleichung

Dieses Übungsblatt umfasst im Wesentlichen die Aufgabe das Poisson Problem in einem sequientiellen FORTRAN Programm zu lösen.

Aufgabe 4A: Erstellen eines Makefiles (60 Punkte)

Die Datei

```
all_in_one.f90
```

soll in fünf separate Dateien zerlegt werden. Es soll ein Makefile erstellt werden, mit dem das Programm kompiliert ($ make) und auch ausgeführt werden kann ($ make run). Alle Abhängigkeiten sollen korrekt berücksichtigt werden. Das heißt, wenn eine Datei neu abgespeichert wird, sollen bei einer erneuten Kompilation mit make nur die Datei selbst und die davon abhängigen Dateien neu übersetzt werden. Die nicht neu übersetzten Programmteile werden beim Linken wieder eingebunden und sollen ein lauffähiges Programm ergeben.

Aufgabe 4B-C: Lösung der Poissongleichung

In den folgenden Aufgaben soll die Poisson-Gleichung gelöst werden, indem zwei unterschiedliche Iterationsverfahren angewendet werden:

- Jacobi-Verfahren
- Gauß-Seidel-Verfahren.

Zu den Programmen soll ein Makefile erstellt werden, das bei Eingabe von "make" beide Programme kompiliert. Bei Eingabe von "make runjakobi" das Jakobi Vefahren und bei Eingabe von "make rungauss" das Gauß-Seidel Verfahren ausführt. Die Eingabe von "make clean" löscht alle kompilierten Teile, so dass nur der Quellcode übrig bleibt.

Für beide Verfahren soll es möglich sein, entweder die Anzahl an Interationen vorzugeben die zu durchlaufen sind, als auch eine Genauigkeit vorzugeben, nach deren Erreichen das Programm terminiert.

Die Poisson-Gleichung ist eine partielle Differentialgleichung der Form:

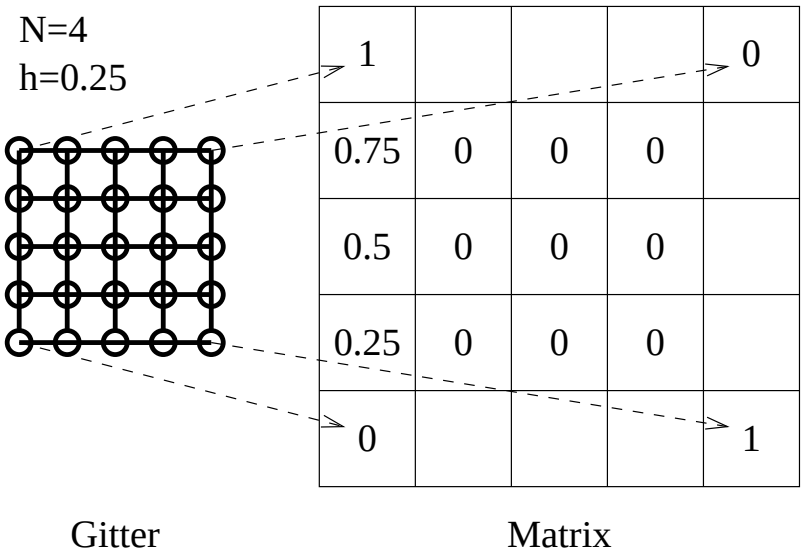
$$-u_{xx}(x,y) - u_{yy}(x,y) = f(x,y) \text{ mit } 0 < x, y < 1 \tag{1}$$

In diesen Aufgaben soll ohne die Berücksichtigung einer Störfunktion gerechnet werden, d.h. wir verwenden $f(x,y) = 0$.

Des weiteren werden die Randwerte vorgegeben mit:

- Randbelegung: $v(1,1) = v(N+1, N+1) = 1$ und $v(1, N+1) = v(N+1, 1) = 0$.
  Alle Randwerte zwischen den Ecken werden linear interpoliert.
  Die inneren Werte der Lösungsmatrix werden auf Null gesetzt.

Die Initialisierung der Rändern ist hier für das Beispiel einer (N+1)x(N+1) Matrix mit N=4 illustriert, die Gitterweite$h$: $1/N$ ist also 0.25:



Ein Überblick über die mathematischen Grundlagen ist im Anhang "Mathematische Grundlagen" gegeben.

Aufgabe 4B: Lösung der Poissongleichung nach Jakobi (180 Punkte)

Im Jakobi Verfahren werden zwei Matrizen verwendet, ähnlich der Vorgehensweise im Game of Life. Die Berechnung der neuen Werte erfolgt nur mit Hilfe der alten Werte. Für die Berechnung mit vorgegebener Anzahl an Iterationen sollen 40 0000 Iterationen durchlaufen werden, während der Abbruch nach Genauigkeit bei einem Unterschreiten der Genauigkeit von 10^{-6} erfolgen soll. Für beide Verfahren sind die Berechnungen in REAL*8 bzw. Double Precission durchzufühen.

Die Ausgabe erfolgt mit Hilfe von sogenannten Interlines mittels einer 9 x 9 Matrix. Die Berechnungs-Matrix b kann damit auf die Ausgabe-Matrix a abgebildet werden.

```
iout = 0
jout = 0
    do i = 1, imax, interlines
          iout = iout+1
             do j =1, jmax, interlines
                 jout = jount+1
                   a(iout,jout) = b(i,j)

             enddo
    enddo
```

Die Interlines bilden ausgewählte Gitterpunkte innerhalb der Matrix ab, mit deren Hilfe das Gesamtverhalten einfach analysiert werden kann. Daher soll zu jedem Lauf die Anfangsbelegung und der Endzustand der Matrix abgebildet werden.

Aufgabe 4C: Lösung der Poissongleichung mittels Gauß-Seidel-Verfahren (180 Punkte)

Das Gauß-Seidel Verfahren verwendet nur eine Matrix. Der aktuell berechnete Wert überschreibt den alten Wert. Für den nächsten Gitterpunkt wird dann sogleich dieser neue Werte als Nachbarpunkt mit verwendet. Für beide Berechnungen, nach Iterationen und Genauigkeit, sollen die gleichen Vorgaben wie beim obigen Jakobi Verfahren (Aufgabe 4B) verwendet werden.

Aufgabe 4D: Leistungsmessungen (120 Punkte)

Abschließend wird eine Leistungsmessung der erstellten Programme durchgeführt. Die Ausgabe erfolgt mit dem TIME Kommando. Die so generierte Ausgabe pro Lauf wird einfach von Hand in eine Datei kopiert. Um ungleiche Auslastung des Knotens auszugleichen sollen pro Aufgabenstellung je 5 Läufe durchgeführt werden. Die Zeiten werden entsprechend in der Datei aufgelistet.

Insgesamt sollen für beide Verfahren (Jakobi und Gauß-Seidel) eine Reihe von Läufen mit der Ausgabe der vom Cluster benötigten Rechnerzeit durchgeführt werden. Die Läufe umfassen die Verwendung einer 97x97 Matrix (Interlines = 11) und einer 185x185 Matrix (Interlines =22). Unter Verwendung des Abbruchkriteriums nach Genauigkeit soll diese sowohl für 10^{-6} und 10^{-8} ausgewiesen werden.
D.h. insgesamt müssen 8 Realisierungen mit jeweils 5 Läufen getestet werden, je Verfahren jeweils beide Matrixen und jeweils beide Abbruchkriterien. Die schnellsten Läufe werden von uns mit Bonuspunkten versehen. Innerhalb aller abgegebenen Zeiten wird der schnellste Lauf mit 30 Punkten, der zweitbeste Lauf mit 20 Punkten und der drittbeste Lauf mit 10 Bonuspunkten versehen.

Abgabe

Die auf dem Cluster lauffähigen FORTRAN Programme sollen als Quellcode mit der Angabe der Gruppe (Personen in der Gruppe) bis zum Mittwoch den 22.5.2012 geschickt werden an:

hermann.lenhart@informatik.uni-hamburg.de

Als Subject im Kopf der Mail bitte die Angabe: PPG-12 Blatt4 und die Liste der Familiennamen der Personen in der Übungsgruppe.

Bitte nur den **FORTRAN Quellcode**, das dazugehörige **Makefile** sowie die **Datei der Leistungsanalyse** schicken, keinen Objektcode oder Executables!

Anhang: Mathematischer Hintergrund

Viele natürliche und technische Vorgänge lassen sich durch partielle Differentialgleichungen beschreiben. Ein Beispiel hierfür ist die Poisson-Gleichung. Mangels vorhandener analytischer Lösungsformeln muss man sich oft Methoden der numerischen Mathematik bedienen.

Hier gelangt man zunächst durch

(i) Diskretisierung (Festlegung der zu berechnenden Punkte im gewünschten Lösungsgebiet) und

(ii) Ersetzen der Differentialquotienten durch Differenzenquotienten

zu einem System von linearen Gleichungen. Für die Berechnung des Lösungsvektors dieses Systems existieren direkte und indirekte Verfahren. Wegen der Nachteile der direkten Verfahren (Eliminationsverfahren wie z. B. die Gauß-Elimination), nämlich hohe algorithmische Komplexität einerseits und numerische Instabilität andererseits. Heute bevorzugt man indirekte Verfahren, zum Beispiel Iterationsverfahren, bei denen man sich iterativ bis zu einer gewünschten Genauigkeit der exakten Lösung annähern kann (sofern das Iterationsverfahren konvergiert). Zwei dieser Iterationsverfahren werden hier kurz vorgestellt.

A: Problemstellung

Gegeben ist eine partielle Differentialgleichung der Form

$$-u_{xx}(x,y) - u_{yy}(x,y) = f(x,y) \text{ mit } 0 < x, y < 1 \tag{2}$$

Diese Darstellung wird als **Poisson-Problem** bezeichnet. Dabei bezeichnet u_{ii} die zweite Ableitung der Funktion u nach i. Die Funktion $f(x,y)$ bezeichnet man als **Störfunktion**.

Die Eckpunkte der Randwerte $u(1,1)$, $u(1,N+1)$, $u(N+1,1)$ und $u(N+1,N+1)$ sind gegeben (siehe obige Abbildung). Die Werte auf dem Rand sollen dazuwischen interpoliert werden. Gesucht wird $u(x,y)$ für $0 < x, y < 1$.

B: Übergang von Differential- zu Differenzenquotienten

Wir definieren die inneren Gitterpunkte

$$u_{i,j} := u(i * h, j * h) \text{ mit } i, j = 2, ..., N \tag{3}$$

Die Ersetzung der partiellen Ableitungen aus (1) durch finite Differenzen zweiter Ordnung:

$$u_{xx;i,j} := 1/h^2(u_{i+1,j} - 2u_{i,j} + u_{i-1,j})$$

$$u_{yy;i,j} := 1/h^2(u_{i,j+1} - 2u_{i,j} + u_{i,j-1})$$

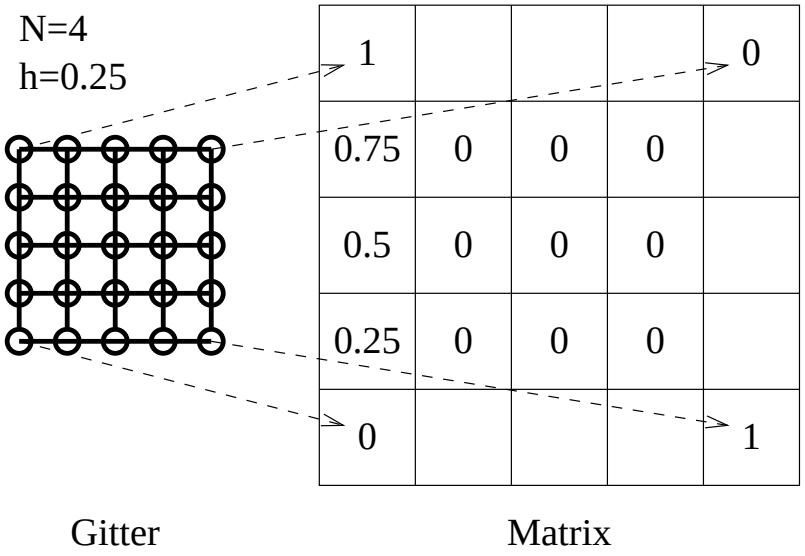
liefert ein **System von linearen Gleichungen** der Form:

$$1/h^2(4v(i,j) - v(i-1,j) - v(i+1,j) - v(i,j-1) - v(i,j+1)) = f(i,j) \tag{4}$$

C: Diskretisierung

Die Lösung der Poisson-Gleichung soll auf dem Gebiet $[0,1] \times [0,1]$ berechnet werden. Einfachste Diskretisierung ist ein **äquidistantes quadratisches Gitter**.

Anzahl Intervalle in jeder Richtung: N
Anzahl Punkte auf Gesamtgebiet (mit Rand): $(N+1)^2$
Anzahl innerer Punkte: $(N-1)^2$
Gitterweite h: $1/N$



Dieses Gitter kann in einer $(N+1) \times (N+1)$-Matrix gespeichert werden. Jeder Eintrag in der Matrix repräsentiert einen Punkt des Gitters. Die Randpunkte des Gitters werden vor Beginn der Berechnung vorbelegt.

D: Schema zum Lösen der Differentialgleichung

Das zentrale Element der Berechnung ist der sogenannte "Abtaststern". Der Stern wird wie folgt berechnet:

$$Star = -v(i, j+1) - v(x-1, j) + 4 * v(i, j) - v(i+1, j) - v(i, j-1)$$

Danach wird der Korrekturwert berechnet. Die allgemeine Beschreibung hierzu für eine Variable $Corr$ lautet:

$$Corr = (f(i, j) * h^2 - Star)/4$$

Mittels des berechneten Korrekturterms wird die neue Iteration t der Matrix $v_t(i, j)$, aus dem alten Wert $v_{t-1}(i, j)$ der vorherigen Iteration t-1 berechnet.

$$v_t(i, j) = v_{t-1}(i, j) + Corr$$

Für die beiden Iterationsverfahren ist folgendes zu beachten:

Jacobi-Verfahren:
Verwendet zwei Matrizen für v: Die Berechnung der neuen Werte erfolgt nur mit Hilfe der alten Werte.

Gauß-Seidel-Verfahren:
Verwendet nur eine Matrix für v, d.h. neue Werte werden verwendet, sobald sie berechnet wurden und gehen damit gleich wieder in die Berechnung des folgenden Sterns ein.

E: Betrachtung zur Genauigkeit der Lösung

Darstellung von (4) in Matrixform: $Au = h^2 f$ (Herleitung und Aufbau der Matrix A soll für die Anwendung des Iterationsverfahrens hier nicht weiter betrachtet werden.)

Exakte Lösung: u
Näherung: v (soll iterativ berechnet werden, bis v nur noch minimal von u abweicht)
Fehler: $e = u - v$
Leider: u ist unbekannt, d.h. e ist nicht feststellbar.
Aber: berechenbar sind

- **Residuum** $r := h^2 f - Av$ (r ist der Betrag, um den die Näherung von v vom Originalproblem $Au = h^2 f$ entfernt ist)
- **Norm des Residuums** $\| r \|_\infty := \max | r(i,j) |$
- Zusammenhang: $\| r \|_\infty = 0 \Longleftrightarrow e = 0$

Aus $Au = h^2 f$ und $Av = h^2 f - r$ erhält man durch Subtraktion $Ae = r$, bzw. $e = A^{-1} r$, genannt **Residuumsgleichung**.

F: Pragmatische Vorgehensweise zum Abbruch nach Genauigkeit

An jedem Gitterpunkt wird die Differenz aus alten und neuen Werten berechnet.

$Diff = v_t(i,j) - v_{t-1}(i,j)$

Unterschreitet diese Differenz $Diff$ an allen Punkten die geforderte Genauigkeit(z.B. 10^{-6}), wir das Programm abgebrochen.

Literatur (Begleitend und weiterführend)

- Stoer, Bulirsch: Einführung in die numerische Mathematik II, Heidelberger-Taschenbücher, Band 114, Springer
- William H. Press: Numerical Recipes in Pascal/C/Fortran, Cambridge, USA 1990