Übungen zur Vorlesung

"Objektorientiertes Programmieren im Wissenschaftlichen Rechnen"

Dr. O. Ippisch, Dr. C. Engwer

Abgabe am 22. 06. 2010 in der Vorlesung

Die Übung am 24. 06. 2010 entfällt. Es wird einen Zettel mit weiterführenden Aufgaben zu diesem Zettel geben. Alle kritischen Teile werden als Musterlösung auf die Vorlesungs-homepage gestellt.

ÜBUNG 1 ABSTRAKTE MATRIXALGORITHMEN: GAUSS-SEIDEL

Auf dem letzten Übungsblatt wurde eine abstrakte Matrix-Schnittstelle basierend auf Iteratoren programmiert. Sollten sie Probleme gehabt haben, finden sie eine Musterlösung auf der Homepage.

Basierend auf dieser Matrixschnittstelle wollen wir nun einen Algorithmus zur iterativen Lösung von Gleichungssystemen programmieren.

Der Gauß-Seidel-Algorithmus liefert für Matrizen, deren deren Spektraradius kleiner als 1 ist, eine näherungsweise Lösung des linearen Gleichungssystems.

Gegeben eine $n \times n$ Matrix A und eine rechte Seite b, suchen wir eine näherungsweise Lösung des Vektors x, so dass

$$Ax = b$$

erfüllt ist. Dies entspricht n Gleichungen eines linearen Gleichungssystems.

Um das System zu lösen, wird die k-te Gleichung nach x_k aufgelöst. Im $m+1$-ten Iterationsschritt ergibt somit folgende Gleichung:

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

Implementieren sie den Gauss-Seidel-Algorithmus basierend auf der abstrakten Iterator-Schnittstelle. Schreiben sie ihr Programm zunächst so, dass eine feste Anzahl an Iterationen durchgeführt wird. Das Interface soll folgendermaßen aussehen:

```
template<class M, class T>
void gaussseidel (const M& A, const std::vector<T> & b, std::vector<T> & x, int maxIter);
```

Hinweis: Testen sie ihr Programm an folgendem Beispiel:

$$\begin{pmatrix} 2 & 1 & 0 \\ 1 & 3 & 2 \\ 0 & 1 & -4 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \\ -0.5 \end{pmatrix}^T = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

10 Punkte

ÜBUNG 2 DÜNNBESETZTE MATRIZEN

In Anwendungen im wissenschaftlichen Rechnen, Matrizen haben oft sehr viele Einträge die 0 sind. Diese sogenannte dünnbesetzte Matrixstruktur ist eine direkte Folge der Diskretisierung. Für solche

dünnbesetzte Matrizen gibt es effiziente Verfahren, um ein Gleichungssystem näherungsweise zu lösen. Der in der vorherigen Übung implementierte Gauss-Seidel Algorithmus ist ein Beispiel dafür.

Um diese Verfahren effizient zu implementieren (sowohl Rechenzeit, als auch Speicherverbrauch) ist es üblich diese dünnbesetzen Strukturen auch bei der Speicherung der Matrix zu berücksichtigen.

In dieser Übung wollen wir eine einfache Variante einer dünnbesetzten (sparse) Matrix implementieren.

- Wir verwenden hierzu den Compressed-Row-Storage Ansatz.
- Diese soll unter der generischen Iterator-Interface von Blatt 8 verfügbar sein: `RowIterator` und `ColIterator`, sowie die dazugehörigen `begin` und `end` Methoden.
- Einträge die 0 sind werden nicht gespeichert und vom Iterator übersprungen (da sie bei Matrix Operationen keinen Beitrag haben).
- Da nur die Non-Zero Einträge gespeichert werden, muss die Anzahl dieser Matrix mitgeteilt werden. Das passiert im Konstruktor:

```
SparseMatrix(unsigned int rows, unsigned int cols, unsigned int nonZeros);
```

- Die Datenstruktur sieht folgender maßen aus (Pseudocode):

```
template<typename T>
SparseMatrix {
    T data[nonZeros];
    unsigned int column[nonZeros];
    unsigned int rowOffset[rows];
};
```

  - Die Daten werden zeilenweise in `data` gespeichert.
  - Zu jedem Non-Zero Eintrag wird im `column` Vektor gespeichert, in welcher Spalte er steht.
  - Der `rowOffset` Vektor speichert für jede Zeile, wo die Daten dieser Zeile im den `data` und `column` Vektoren beginnen.

  Beispiel (Matrix aus Aufgabe 1):

```
rows: 3
cols: 3
nonZeros: 7
data: 2, 1, 1, 3, 2, 0, -4
column: 0, 1, 0, 1, 2, 1, 2
rowOffset: 0, 2, 5
```

- Testen Sie Ihre Implementierung mit den beiden Matrix-Algorithmen aus Blatt 8 und Blatt 9. Verwenden Sie die beschriebenen Testfälle.

14 Punkte

Wie immer gilt: Kommentieren Sie Ihr Programm. Erklären Sie was Sie tuen.