Anwesenheitsübung 8 zur Vorlesung ‘Numerische Methoden der Physik’ SS 2014

Bastian Knippschild, Christian Jost und Mitarbeiter

Bearbeitung in den Übungen am 03. – 05. 06. 2014

Dieses Aufgaben knüpft direkt an die letzte Übung an, in der der einfachste Fall des Algorithmus zur Gauss-Elimination implementiert werden sollte.

Sollten Sie beim letzten Mal mit der Implementierung der Gauss-Elimination ohne Vertauschungen nicht fertig geworden sein, beenden Sie zuerst diese Aufgabe, bevor Sie sich den neuen Aufgaben zuwenden.

Matrix-Matrix-Multiplikation

In dieser ersten Aufgabe soll eine Funktion implementiert werden, die zwei quadratische Matrizen X und Y multipliziert. Testen Sie Ihre Funktion, in dem sie $A = L \cdot R$ für

$$L = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 \\ 3 & \frac{4}{3} & 1 & 0 \\ 4 & \frac{5}{3} & \frac{5}{4} & 1 \end{pmatrix}, \quad R = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 0 & -3 & -4 & -5 \\ 0 & 0 & -\frac{8}{3} & -\frac{10}{3} \\ 0 & 0 & 0 & -\frac{5}{2} \end{pmatrix}$$

berechnen und überprüfen, dass die zurückgegebene Matrix A tatsächlich mit der folgenden Matrix übereinstimmt

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 2 & 3 \\ 3 & 2 & 1 & 2 \\ 4 & 3 & 2 & 1 \end{pmatrix}$$

Lösung linearer Gleichungssysteme

Als weiterer Schritt auf dem Weg zur LR-Zerlegung soll in dieser Aufgabe ein Algorithmus implementiert werden, der, bei vorgegebener Zerlegung $A = L \cdot R$, den Lösungsvektor des zugehörigen LGS $A \cdot x = b$ bestimmt. Dies wird in zwei Schritten erreicht:

a) Lösen von $L \cdot c = b$ durch Vorwärtssubstitution

b) Lösen von $R \cdot x = c$ durch Rückwärtssubstitution

Für den zweiten Schritt sollten Sie ihren bereits vorhandenen Code aus dem letzten Tutorium wiederverwenden können. Testen Sie ihren Algorithmus für die oben angegebenen Matrizen L, R und den Vektor $b = (3.0, 2.0, 3.0, 4.0)^T$. Als Lösung sollten Sie $x = (0.2, 1.0, 0.0, 0.2)^T$ erhalten.

LR-Zerlegung ohne Vertauschungen

Nun soll die LR-Zerlegung ohne Vertauschungen implementiert werden. Für eine $n \times n$-Matrix A können Sie von dem folgenden Pseudocode ausgehen:

```
for i = 0,...,n-1 do
  for j = i,...,n-1 do
    for k = 0,...,i-1 do
      A[i][j] = A[i][j] - A[i][k] * A[k][j]
    end for
  end for
  for j = i+1,...,n-1 do
    for k = 0,...,i-1 do
      A[j][i] = A[j][i] - A[j][k] * A[k][i]
    end for
    A[j][i] = A[j][i] / A[i][i]
  end for
end for
return A
```

Verwenden Sie so viel von Ihrem bestehendem Code, wie möglich. Testen Sie Ihr Programm an der oben angegebenen Matrix A. Sie sollten als Ergebnis die oben angegebenen Matrizen R und L erhalten.

LR-Zerlegung mit Pivotisierung

In dieser letzten Aufgabe soll nun die vollständige LR-Zerlegung inklusive Pivotisierung implementiert werden. Betrachten Sie dazu den folgenden in der Vorlesung diskutierten Pseudocode:

```
for i = 0,...,n-1 do
  l = index ( max_{m=i,...,n-1} |A[m][i]| / summe_{j=i,...,n-1} |A[m][j]| )
  Speichere i<-->l in der Liste der Vertauschungen sigma
  for j = i,...,n-1 do
    for k = 0,...,i-1 do
      A[i][j] = A[i][j] - A[i][k] * A[k][j]
    end for
  end for
  for j = i+1,...,n-1 do
    for k = 0,...,i-1 do
      A[j][i] = A[j][i] - A[j][k] * A[k][i]
    end for
    A[j][i] = A[j][i] / A[i][i]
  end for
end for
return L,R,sigma
```

Überlegen Sie sich zunächst wie Sie die Vertauschungen sinnvoll realisieren. Auch bei dieser Aufgabe sollten Sie möglichst viel von Ihrem Code wiederverwenden. Überprüfen Sie sorgfältig, dass

einzelne Programmteile korrekt funktionieren, insbesondere die Bestimmung des Index l. Ihr Programm sollte anschließend für (im Prinzip) beliebige $n \times n$-Matrizen funktionieren.

Ihren fertigen Algorithmus können Sie an dem folgenden Gleichungssystem testen:

$$A = \begin{pmatrix} 4 & -2 & 1 & -1 \\ -2 & 3 & 4 & -5 \\ 6 & -3 & 5 & -1 \\ -5 & 6 & 0 & 2 \end{pmatrix}, \quad b = \begin{pmatrix} -1 \\ -4 \\ 11 \\ 15 \end{pmatrix} \tag{1}$$

Als Lösungsvektor des LGS $Ax = b$ sollten Sie $(1.0, 2.0, 3.0, 4.0)^T$ erhalten.