Universität Bielefeld	Computerphysik	Prof. Dr. Jürgen Schnack
Fakultät für Physik	SS 2016	jschnack@uni-bielefeld.de

Aufgabenblatt 12

12.1 Lineare Gleichungssysteme (Hausaufgabe + Email an die Tutoren, Einsendeschluss Mitternacht vor der Übung)

In der Vorlesung wurde die Bibliothek `LAPACK` besprochen. Aus dieser Bibliothek haben wir die Unterroutine `DGESV` verwendet, um lineare Gleichungssysteme der Form

$$A \cdot X \;=\; B \tag{1}$$

zu lösen. Dabei ist A eine Matrix, B und X haben die gleiche Dimension, sind also z.B. Vektoren oder Matrizen gleicher Größe.

1. Informieren Sie sich über die Bibliothek `LAPACK`. Dazu können Sie zum Beispiel bei `www.netlib.org/lapack` nachschauen. Informieren Sie sich auch über die Parameter, die an die Routine `DGESV` übergeben werden müssen. Entsprechende Informationen finden Sie unter „Individual Routines“.

   Bei `LAPACK` handelt es sich um eine Bibliothek, die ursprünglich in Fortran 77 geschrieben und dann später für andere Programmiersprachen portiert wurde. Informieren Sie sich unter `www.netlib.org/clapack/readme` darüber, was alles zu beachten ist, wenn eine `LAPACK`-Funktion innerhalb eines C-Programms aufgerufen werden soll. Wichtig ist insbesondere der Unterschied zwischen Fortran und C bei der Speicherung zweidimensionaler Felder („column-major order“ im Gegensatz zu „row-major order“). Wiederholen Sie, wie man auf die Komponenten einer Matrix statt (wie gewohnt) mit zwei Indizes mit einem linearen Index zugreifen kann.

2. Erstellen Sie ein Programm in C/C++ (oder der Programmiersprache Ihrer Wahl), das die Matrizen A und B einliest und das lineare Gleichungssystem durch Aufruf der Routine `DGESV` löst. Verwenden Sie die auf meiner Webseite hinterlegten Matrizen.

   Wenn Sie zur Lösung der Aufgabe ein C-Programm auf einer Linux-Maschine schreiben, dann ist (neben den obigen Hinweisen) Folgendes zu beachten. Bis jetzt haben Sie nur einfache „externe“ Funktionen wie z.B. `gaussLegendre.h` kennengelernt, deren Source-Code in das Hauptprogramm durch einen `#include`-Befehl eingebunden werden musste, um die Funktion nutzen zu können. Kompliziertere Bibliotheken (wie z.B. `LAPACK`) liegen dagegen typischerweise in Form von bereits kompilierten Objekt-Dateien (Endung z.B. `.a` oder `.so`) vor. Die zugehörige Header-Datei (Endung `.h`) enthält dann nicht den kompletten Source-Code (bei kommerziellen Bibliotheken werden Sie diesen niemals zu Gesicht bekommen), sondern nur Deklarationen der Funktionen, d.h. die benötigten Informationen über Rückgabewerte und Argumente der Funktionen.

   Um in Ihrem Programm `LAPACK`-Funktionen verwenden zu können, müssen Sie zunächst die Header-Dateien `f2c.h` und danach (!) `clapack.h` einbinden. Beim

Kompilieren müssen dann die drei Bibliotheken `liblapack.a`, `libblas.a` und `libf2c.a` korrekt in Ihr Programm integriert („gelinkt“) werden. Kopieren Sie dazu die drei Dateien in dasselbe Verzeichnis, in dem sich auch Ihr Programm befindet. Kompilieren Sie das Programm dann folgendermaßen:

```
gcc -o <Name> <Source Code> -L. -llapack -lblas -lf2c -lm -Wall
```

Fortgeschrittene können auch die zentral gespeicherten Bibliotheken (meist irgendwo in `/usr/lib`) nutzen und den Compiler-Befehl entsprechend abändern.

3. Wie kann man das Inverse einer Matrix bestimmen?

4. Schreiben Sie selbst ein Programm, das für eine beliebige Matrix A eine LU-Zerlegung durchführt. Testen Sie es anhand des Beispiels. Welchen Vorteil bietet eine LU-Zerlegung im Vergleich zu anderen Methoden?

5. Bestimmen Sie die Determinante der Koeffizientenmatrix A.

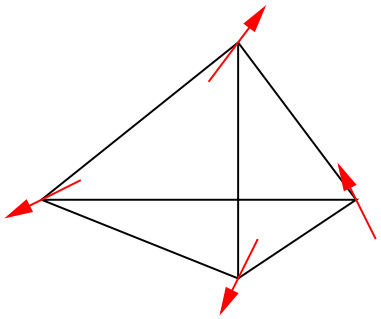
Wenn Sie irgendwo auf Schwierigkeiten stoßen, geben Sie nicht auf! Schreiben Sie den Tutoren eine Email. Die LAPACK werden Sie u.U. in Ihrem Studium oder Berufsleben noch öfter brauchen. Das sollten Sie einmal durchlitten und verstanden haben.

Wenn Sie LAPACK erfolgreich installiert und in Ihr Programm eingebunden haben, teilen Sie bitte Ihre Erfahrungen unter stud.ip im wiki. Vielen Dank.

WINDOWS-Nutzer sollen hier nachschauen:
`www.netlib.org/clapack/`
Da gibt es *prebuilt libraries*.

12.2 Eigenwerte (Hausaufgabe + Bearbeiten in den Übungen)

Als ein realistisches Eigenwertproblem betrachten wir im folgenden das Heisenberg-Modell, in dem die magnetischen Eigenschaften eines Spinsystems beschrieben werden sollen. Unser Beispielsystem besteht aus 4 Spins, von denen jeder mit allen anderen mit gleicher Stärke wechselwirkt. Die Spins haben dabei die Quantenzahl $s = 5/2$ (kleines Problem) bzw. $s = 7/2$ (großes Problem).



Der Hamiltonoperator für dieses Spinsystem lautet

$$\underset{\sim}{H} \quad = \quad \frac{2J}{\hbar^2} \sum_{i<j} \underset{\sim}{\vec{s}}_i \cdot \underset{\sim}{\vec{s}}_j \quad , \quad J > 0 \; . \tag{2}$$

Dabei nummerieren i und j die Spins, und J ist die Stärke der Wechselwirkung (Austauschkopplung). Ein freundlicher Kommilitone hat für diesen Hamiltonoperator in einer Orthonormalbasis seiner Wahl die Matrixelemente von $\underset{\sim}{H}/J$ ausgerechnet (lineare Dimension $(2s+1)^4$). Diese stehen als Datei auf meiner Webseite bereit.

1. Informieren Sie sich auf `www.netlib.org/lapack` über die Parameter, die an die Eigenwertroutine `DSYEV` übergeben werden müssen. Entsprechende Informationen finden Sie wieder unter „Individual Routines“.

2. Erstellen Sie ein Programm in C/C++ (oder der Programmiersprache Ihrer Wahl), das die zu diagonalisierende Matrix einliest und die Eigenwerte durch Aufruf der Routine `DSYEV` bestimmt.

   Um in Ihrem Programm `LAPACK`-Funktionen verwenden zu können, müssen Sie (auf srv-semd01) zunächst wieder die Header-Dateien `f2c.h` und danach `clapack.h` einbinden. Beim Kompilieren müssen dann die drei Bibliotheken `liblapack.a`, `libblas.a` und `libf2c.a` korrekt in Ihr Programm integriert („gelinkt“) werden. Kopieren Sie dazu wie zuvor die drei Dateien in dasselbe Verzeichnis, in dem sich auch Ihr Programm befindet, und kompilieren Sie das Programm mit:

   ```
   gcc -o <Name> <Source Code> -L. -llapack -lblas -lf2c -lm -Wall
   ```

3. In diesem Spezialfall kann man die numerisch erhaltenen Eigenwerte mit einer analytischen Lösung vergleichen. Gemäß dieser treten nur die folgenden Eigenwerte auf:

$$E(S) \quad = \quad J\left[S(S+1) - 4s(s+1)\right] \; . \tag{3}$$

Dabei ist $S = 0, 1, 2, \ldots, 4s$ die Quantenzahl des Gesamtspins. Die Eigenwerte sind oft mehrfach entartet. Vergleichen Sie Ihre Eigenwerte mit dem analytischen Ergebnis.

4. Berechnen Sie die innere Energie $U(T)$ und die Wärmekapazität $C(T)$ als Funktion der Temperatur T und stellen Sie diese graphisch dar. Die Größen sind wie folgt definiert:

$$Z(T) = \sum_i \exp\left\{-\frac{E_i}{k_B T}\right\} , \tag{4}$$

$$U(T) = \frac{1}{Z(T)} \sum_i E_i \exp\left\{-\frac{E_i}{k_B T}\right\} , \tag{5}$$

$$C(T) = \frac{k_B}{(k_B T)^2} \left[\frac{1}{Z(T)} \sum_i E_i^2 \exp\left\{-\frac{E_i}{k_B T}\right\} - U^2(T)\right] . \tag{6}$$

Führen Sie diese Rechnungen einheitenlos durch. Die Matrixelemente sind ja schon einheitenlos gegeben. Nutzen Sie analog E_i/J, U/J, $k_B T/J$ und C/k_B als einheitenlose Größen (Das bedeutet nichts anderes, als dass Sie in den Rechnungen $k_B = 1$ setzen.). Verwenden Sie für die Berechnung der Exponentialfunktionen den Trick aus Aufgabe 4.4, d.h. verwenden Sie statt der absoluten Energien die Anregungsenergien.

5. Lassen Sie von der Routine `DSYEV` auch die Eigenvektoren berechnen. Überprüfen Sie exemplarisch für zwei Vektoren Ihrer Wahl, ob sie orthonormal sind.