Übungen zur Vorlesung „Physik auf dem Computer"
Sommersemester 2018

Übungsgruppenleiter:
Robin Bardakcioglu – `rhb@itp1.uni-stuttgart.de; Do. 14:00 - 15:30 Uhr, 5.331`
Johannes Reiff – `jreiff@itp1.uni-stuttgart.de; Di. 14:00 - 15:30 Uhr, 4.141`
Matthias Feldmaier – `fem@itp1.uni-stuttgart.de; Do. 14:00 - 15:30 Uhr, 4.141`

Übungsblatt 11 Ausgabe: 04. Juli 2018

Dienstagsübung: schriftliche Abgabe 08.07.18, Besprechung 10.07.18
Donnerstagsübungen: schriftliche Abgabe 10.07.18, Besprechung 12.07.18

Aufgabe 26: Jacobi-Rotation

Eine reelle symmetrische Matrix lässt sich, wie in der Vorlesung erläutert wurde, durch Jacobi-Rotationen auf Diagonalgestalt bringen. In einem Schritt $\boldsymbol{A}' = \boldsymbol{R}_{pq}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{R}_{pq}$ setzt man durch die Operationen

$$\begin{aligned}
&a'_{kp} = c a_{kp} - s a_{kq} \,, \qquad a'_{kq} = c a_{kq} + s a_{kp} \,, \qquad \text{für alle } k \neq p, q \,, \\
&a'_{pp} = c^2 a_{pp} + s^2 a_{qq} - 2cs a_{pq} \,, \qquad a'_{qq} = s^2 a_{pp} + c^2 a_{qq} + 2cs a_{pq} \,, \\
&a'_{pq} = (c^2 - s^2) a_{pq} + cs(a_{pp} - a_{qq})
\end{aligned}$$

(und identischen für die Elemente a'_{pk}, a'_{qk}, a'_{qp}) die beiden Matrixelemente a'_{pq}, a'_{qp} auf Null. Dabei gilt $c = \cos(\alpha)$ und $s = \sin(\alpha)$ mit dem Rotationswinkel aus der Vorlesung, d.h.

$$t = \tan(\alpha) = \frac{\operatorname{sign}\left(\frac{a_{qq}-a_{pp}}{2a_{pq}}\right)}{\left|\frac{a_{qq}-a_{pp}}{2a_{pq}}\right| + \sqrt{\left(\frac{a_{qq}-a_{pp}}{2a_{pq}}\right)^2 + 1}} \,, \qquad c = \frac{1}{\sqrt{1+t^2}} \,, \qquad s = \frac{t}{\sqrt{1+t^2}} \,.$$

a) Laden Sie aus dem Webangebot zur Vorlesung das Programmpaket zu dieser Aufgabe und öffnen Sie die Datei `Eine_Transformation.cc`. Sie erstellt eine mit Zufallszahlen gefüllte symmetrische Matrix, auf die *eine* Jacobi-Rotation für das Element $p = 0, q = 1$ angewandt wird. Zusätzlich wird die Transformationsmatrix abgespeichert, indem die Operation $\boldsymbol{R}_{pq}\boldsymbol{E}$ auf die Einheitsmatrix $\boldsymbol{E}$ angewandt wird. Der eigentliche Algorithmus zur Jacobi-Rotation befindet sich in der Datei `Jacobi.cc` und ist lückenhaft.

ToDo: Beseitigen Sie die Lücken und ergänzen Sie den Algorithmus zur Jacobi-Rotation der Matrix $\boldsymbol{A}$ sowie das Erstellen der Transformationsmatrix. Testen Sie anhand der vorbereiteten Ausgaben für verschiedene Matrixgrößen und Elemente p, q, ob Ihr Algorithmus funktioniert. Compilieren können Sie mit dem Aufruf von `make`.

(4 Punkt(e), Votier)

b) Legen Sie eine Kopie des Programms an.

ToDo: Schreiben Sie diese so um, dass sie eine vollständige Diagonalisierung durchführt. Dafür müssen Sie alle Außerdiagonalelemente $q > p$ (oder umgekehrt) nacheinander so oft durchgehen, bis eine akzeptable Diagonalgestalt erreicht ist. Definieren Sie eine akzeptable Diagonalgestalt durch

$$2\sum_{j>i} a_{ij}^2 < 10^{-10} \sum_i a_{ii}^2 \; .$$

(3 Punkt(e), Votier)

c) Ihr Programm berechnet sowohl die Eigenwerte als auch die Eigenvektoren der Matrix $\boldsymbol{A}$. Überlegen Sie sich, wo die Information abgespeichert ist. Um das Programm zu testen, können Sie folgendes implementieren: Extrahieren Sie einen der Eigenvektoren, multiplizieren Sie ihn mit der Matrix und teilen Sie den erhaltenen Vektor durch den zugehörigen Eigenwert.

ToDo: Geben Sie für eine 5×5-Matrix diesen Vektor zusammen mit dem ursprünglichen Eigenvektor aus. Sie sollten identisch sein.

(3 Punkt(e), Votier)

Aufgabe 27: Lanczos-Algorithmus

Eine Erweiterung der einfachen Vektoriteration aus Aufgabe 14 ist der Lanczos-Algorithmus.

a) Wir betrachten die Folge von Vektoren

$$\mathbf{r}^{(0)}, \mathbf{A}\mathbf{r}^{(0)}, \mathbf{A}^2\mathbf{r}^{(0)}, \mathbf{A}^3\mathbf{r}^{(0)}, \ldots, \mathbf{A}^p\mathbf{r}^{(0)}$$

mit $\mathbf{r}^{(0)}$ Zufallsvektor. Für $p < N$ sind diese Vektoren in der Regel linear unabhängig, aber nicht orthogonal. Der Lanczos-Algorithmus erlaubt jedoch, ausgehend von einem Zufallsvektor $|y_0\rangle$, die Konstruktion eines Satzes orthogonaler Vektoren $|y_n\rangle$, die als Lanczos-Vektoren bezeichnet werden, mit der Iterationsvorschrift ($n \geq 0$, $b_0 \equiv 0$)

$$|y_{n+1}\rangle = \mathbf{A}|y_n\rangle - a_n|y_n\rangle - b_n^2|y_{n-1}\rangle \; . \tag{1}$$

Die Koeffizienten a_n und b_n^2 sind gegeben durch

$$a_n = \frac{\langle y_n|\mathbf{A}|y_n\rangle}{\langle y_n|y_n\rangle} \quad , \quad b_n^2 = \frac{\langle y_n|y_n\rangle}{\langle y_{n-1}|y_{n-1}\rangle} \; . \tag{2}$$

ToDo: Laden Sie sich im Webangebot das Program *lanczos.cpp* runter und lesen Sie sich den Code aufmerksam durch. In der Datei ist eine Funktion definiert, welche einen Inputvektor mit der Matrix **A** multipliziert und das Ergebnis in einen Outputvektor schreibt. Vervollständigen Sie die Schleife in der Funktion *lanczosMethod*, sodass die Koeffizienten a_n und b_n^2 berechnet werden und in die dazu bereitgestellten Datenstrukturen schreibt. Stellen Sie Ihren Quellcode in der Übungsgruppe vor. Beachten Sie, dass

bei der Iteration nicht mehr als drei Lanczos-Vektoren gleichzeitig im Speicher gehalten werden müssen. Erinnern Sie sich hierzu auch an die Verwendung der Eigenbibliothek aus Blatt 05.

(5 Punkt(e), Votier)

b) Wie in der Vorlesung gezeigt, nimmt die Matrix $\mathbf{A}$ im Unterraum der (normierten) Lanczos-Vektoren eine tridiagonale Gestalt an:

$$\tilde{\mathbf{A}}_{ij} = \frac{\langle y_i|\mathbf{A}|y_j\rangle}{||y_i|| \cdot ||y_j||} = \begin{pmatrix} a_0 & b_1 & & & & \\ b_1 & a_1 & b_2 & & & \\ & b_2 & a_2 & b_3 & & \\ & & \ddots & \ddots & \ddots & \\ & & & b_{p-2} & a_{p-2} & b_{p-1} \\ & & & & b_{p-1} & a_{p-1} \end{pmatrix} \tag{3}$$

Dabei ist zu beachten, dass die neue Matrix die Form $(p \times p)$ hat und somit potenziell signifikant kleiner sein kann als $\mathbf{A}$. Für Symmetrische tridiagonale Matritzen lassen sich mit Hilfe direkter Verfahren (QR-Algorithmus) sehr effektiv diagonalisieren.

ToDo: Vervollständigen Sie alle Funktionen im vorherigen Aufgabenteil bereitgestellten Programm und bestimmen Sie den Betragmäßig größten Eigenwert der Matrix $\tilde{\mathbf{A}}$ für $p = 5..200$. Beachten Sie, dass das Programm eventuell den C++11 Standard benötigt, um erfolgreich zu Kompilieren. Stellen Sie Ihr Ergebnis in der Übungsgruppe vor. Wieviele Lanczos-Vektoren sind erforderlich, bis der größte Eigenwert konvergiert? Vergleichen Sie mit der Zahl der Iterationen bei der einfachen Potenzmethode in Aufgabe 14.

(10 Punkt(e), Votier)