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 13 Ausgabe: 11. Juli 2018

Dieses Blatt enthält Beispielaufgaben zur Klausurvorbereitung. Die angegebenen Punkte dienen zur Orientierung und sind **nicht** relevant für den Übungsschein.

Aufgabe 1: Gewöhnliche Differentialgleichungen (1 Punkt)
Gegeben sei das Differentialgleichungssystem erster Ordnung

$$\begin{aligned} \dot{x} &= y\,, \\ \dot{y} &= \frac{\gamma}{x}y + e^x\,. \end{aligned}$$

Formulieren Sie dieses Differentialgleichungssystem als Differentialgleichung zweiter Ordnung.

Aufgabe 2: Gewöhnliche Differentialgleichungen (6 Punkte)
Leiten Sie die sogenannte Verlet-Methode zum numerischen Annähern der Bewegungsgleichung her. Entwickeln sie dazu die Position $x(t)$ an der Stelle t und werten Sie das Taylorpolynom an den Stellen $t+h$ und $t-h$ bis zur vierten Ordnung aus. Geben Sie daraus eine Gleichung zur numerischen Berechnung der Position zum Zeitpunkt $t+h$ an. Worin unterscheidet sich diese Methode grundlegend von einer Runge-Kutta-Methode? Beschreiben Sie, wie mit Hilfe der Gleichung die Bewegungsgleichung angenähert werden kann. Welche Fehlerordnung hat der Algorithmus?

Aufgabe 3: Polynominterpolation (3 Punkte)
Berechnen Sie den Wert des interpolierenden Polynoms 2. Grades mit den Stützstellen $(x, y) \in \{(-1, 52), (0, 17), (2, 127)\}$ an der Stelle $x = 1$ mit Hilfe des Neville-Schemas.

Aufgabe 4: Numerische Integration (4 Punkte)
Schreiben Sie eine C++ oder Pythonfunktion `midpoint(f,a,b,N)`, die die zusammengesetzte Mittelpunktsregel implementiert und das Integral $\int_a^b f(x)dx$ mittels N äquidistanter Stützstellen approximiert.

Aufgabe 5: Numerische Integration (1 Punkt)

Die sogenannte Zustandssumme eines Zweiteilchensystems in einer Box sei

$$\int_0^L \int_0^L \int_0^L \int_0^L \int_0^L \int_0^L \exp\left\{-\beta \frac{1}{2}[(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2]\right\} \mathrm{d}x_1 \mathrm{d}y_1 \mathrm{d}z_1 \mathrm{d}x_2 \mathrm{d}y_2 \mathrm{d}z_2 \,.$$

Welche Methode zur numerischen Integration würden Sie zur Berechnung dieses Integrals benutzen, und warum?

Aufgabe 6: Numerische Integration (3 Punkte)

Skizzieren Sie (graphisch oder verbal) zwei Methoden, wie man die Kreiszahl π numerisch annähern kann.

Aufgabe 7: Zufallszahlen (1 Punkt)

Nennen Sie einen Vor- und einen Nachteil von Pseudozufallszahlen gegenüber „echten“ Zufallszahlen.

Aufgabe 8: Test von Zufallszahlen (1 Punkt)

Nennen Sie drei Methoden, um die Qualität von Zufallszahlen zu überprüfen.

Aufgabe 9: Randomwalk (5 Punkte)

Nach einer durchgezechten Nacht implementiert ein Student den zweidimensionalen Randomwalk wie folgt (randint(a,b) liefert eine Zufallszahl zwischen a und b inklusive der Grenzen):

```
u = randint(0, 3)
if u == 0: x += 1
elif u == 1: x -= 1
elif u == 3: y += 1
elif u == 4: y -= 1
```

Der Algorithmus ist fehlerhaft. Wie groß sind die mittleren Abweichungen vom Ursprung $\langle|x|\rangle$ und $\langle|y|\rangle$ nach 200 bzw. 800 Schritten? Skizzieren Sie die Graphen der mittleren Abweichungen vom Ursprung über der Anzahl Schritte.

Aufgabe 10: Fraktale (3 Punkte)
Wie lässt sich mittels der Box-counting Methode die Hausdorff-Dimension eines Fraktals berechnen? Bestimmen Sie die Hausdorff-Dimension der Kochschen Kurve (siehe Abbildung 1).

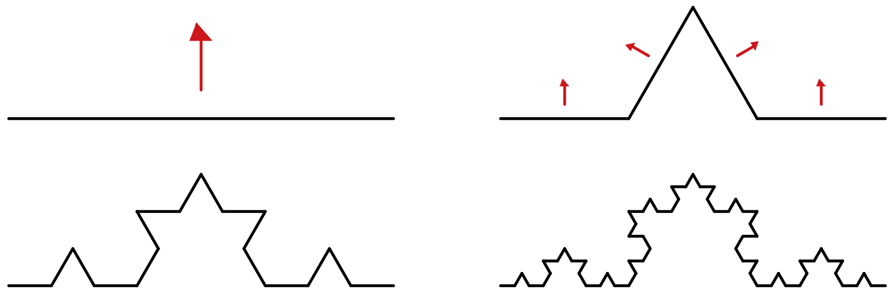

Abbildung 1: Die ersten Schritte bei der Konstruktion der Kochschen Kurve.

Aufgabe 11: Datenanalyse (4 Punkte)
Schreiben Sie eine C++ oder Pythonfunktion (ohne Verwendung von NumPy-Funktionen), die den Mittelwert `mean` und die Varianz `var` einer in einem eindimensionalen Array `x` gegebenen Datenreihe berechnet und zurück gibt.

Aufgabe 12: Fouriertransformation (3 Punkte)
Skizzieren Sie graphisch die Fouriertransformierte der Funktion $f(x) = \sin \pi x + \sin 2\pi x$. Bitte denken Sie daran, die Achsen zu beschriften!

Aufgabe 13: Fouriertransformation (3 Punkte)
Skizzieren Sie (graphisch oder verbal) die Bedeutung der Nyquist-Frequenz eines Signals.

Aufgabe 14: Schnelle Fouriertransformation (1 Punkt)
Nennen Sie einen Vor- und einen Nachteil der „Schnellen Fouriertransformation“ (FFT) gegenüber der diskreten Fouriertransformation (DFT).

Aufgabe 15: Kreuzkorrelationsfunktionen (3 Punkte)
Wie lässt sich die Kreuzkorrelationsfunktion zwischen zwei (reellen) Signalen

$$c(t) = (f \star g)(t) = \int_{-\infty}^{+\infty} f(\tau) g(\tau + t)\, \mathrm{d}\tau$$

mit Hilfe von Fouriertransformationen berechnen? Skizzieren Sie die Kreuzkorrelationsfunktion der beiden in Abbildung 2 gezeigten Signale.

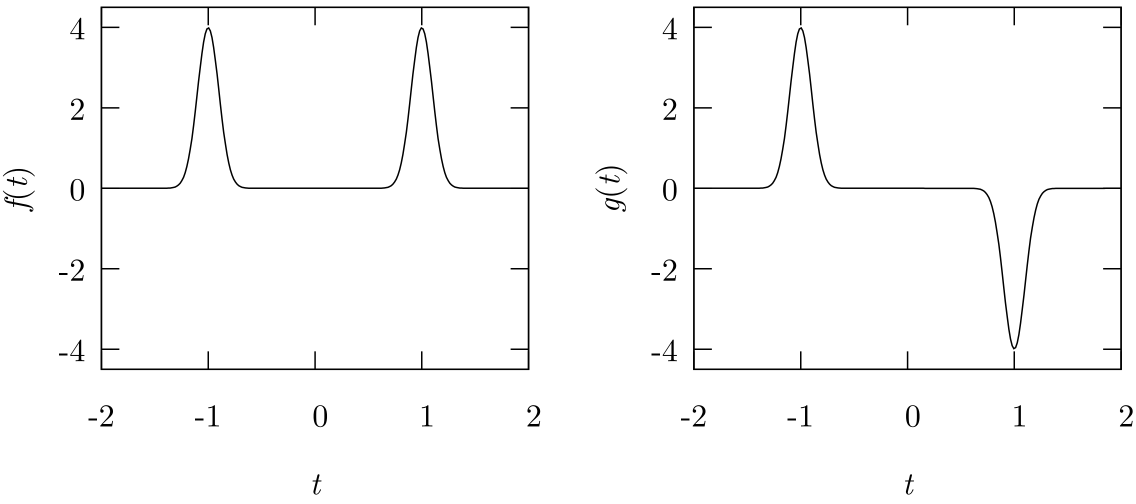

Abbildung 2: Die Funktionen $f(t)$ und $g(t)$.

Aufgabe 16: Hochauflösende Signalanalyse (6 Punkte)
Für ein Signal der Form

$$f_n = \sum_{k=1}^{2} d_k\, z_k^n \tag{1}$$

seien die Signalpunkte $f_1 = -4$, $f_2 = 6$, $f_3 = -10$ und $f_4 = 18$ gegeben. Bestimmen Sie in den folgenden drei Schritten die $\{z_k, d_k\}$ für $k = 1, 2$ mittels hochauflösender Signalanalyse.

1. Schritt:
Schreiben Sie die "linear predictor" Gleichungen

$$f_n = \sum_{k=1}^{2} a_k\, f_{n+k}$$

für $n = 1, 2$ in Matrix-Vektor-Form und bestimmen Sie die a_k mittels Gaußelimination.

2. Schritt:
Bestimmen Sie die z_k als Nullstellen des Polynoms

$$\sum_{m=1}^{2} a_m\, z^m - 1 = 0\,.$$

3. Schritt:
Bestimmen Sie nun die Amplituden d_k aus dem linearen Gleichungssystem (1) für $n = 1, 2$.

Aufgabe 17: Matrixinversion (3 Punkte)
Invertieren Sie die folgende Matrix mit Hilfe der Gaußelimination:

$$\begin{pmatrix} 1 & 5 \\ 2 & 11 \end{pmatrix}$$

Aufgabe 18: Eigenwertgleichungen (1 Punkt)
Beschreiben Sie die Vor- und Nachteile (a) direkter und (b) iterativer Verfahren zur numerischen Lösung von Eigenwertgleichungen der Form $A\boldsymbol{x} = \lambda\boldsymbol{x}$.

Aufgabe 19: Jacobi-Verfahren (5 Punkte)
Wieviele elementare Jacobi-Rotationen sind erforderlich zur Diagonalisierung einer reell-symmetrischen (2×2)-Matrix? Bestimmen Sie mit dem Jacobi-Verfahren die Eigenwerte und Eigenvektoren der Matrix

$$A = \begin{pmatrix} 5 & 1 \\ 1 & 5 \end{pmatrix}$$

indem Sie A in der Form $A = ODO^\mathsf{T}$ mit Orthogonalmatrix O und Diagonalmatrix D schreiben. Benutzen Sie die Formel

$$\tan 2\phi = \frac{2a_{pq}}{a_{qq} - a_{pp}}$$

für die Drehwinkel ϕ der elementaren Jacobi-Rotationen.

Aufgabe 20: Jacobi-Verfahren (3 Punkte)

Die folgende C++ Funktion soll das Jacobi-Verfahren implementieren, ist aber fehlerhaft. Korrigieren Sie die drei Fehler (gerne auch direkt im Code). Diese sind sowohl logischer als auch syntaktischer Natur.

```cpp
#include <cassert>
#include <iostream>

#include <Eigen/Dense>


Eigen::VectorXd jacobi(
    const Eigen::MatrixXd& a, const Eigen::VectorXd& b,
    Eigen::VectorXd x0, double eps = 1e-6
)
{
    assert(a.rows() == b.size());
    assert(a.cols() == b.size());
    assert(b.size() == x0.size());

    while(true)
    {
        Eigen::VectorXd x = b
        for(int i = 0; i < x.size(); ++i)
        {
            for(int j = 0; j < x.size(); ++j)
            {
                if(i != j)
                    x[i] += a(i, j) * x0[j];
            }
        }

        x[i] /= a(i, i);

        if((x0 - x).cwiseAbs().maxCoeff() < eps)
            return x;

        x0 = x;
    }
}
```