Universität Stuttgart – Institut für Computerphysik

Klausur

Physik auf dem Computer SS 2013

JP Dr. Axel Arnold Dr. Olaf Lenz Tobias Richter
Elena Minina

7 August 2013

Name	
Vorname	
Matrikelnummer	

Hinweise

- In der Regel gibt der verfügbare freie Platz einen Hinweis darauf, welchen Umfang die Lösung haben sollte.
- Die Antworten sind, soweit möglich, ganzzahlig oder Brüche mit kleinen Nennern.
- Lies Dir *alle* Fragen am Anfang durch, bevor Du anfängst, sie zu beantworten.
- Falls der Platz nicht ausreichen sollte, verwende zusätzliche Blätter. Beschrifte diese unbedingt mit Deinem Namen und Matrikelnummer!
- Die Maximalpunktzahl ist 100.
- Zum Bestehen der Klausur sind 50 Punkte notwendig.

Viel Erfolg!

1 Lineare Algebra I (10 Punkte)

Aufgabe 1: (1 Punkt)

Welches numerische Verfahren kannst Du zum exakten Lösen eines linearen Gleichungssystems verwenden? Nenne zwei Varianten.

Antwort:

Aufgabe 2: (4 Punkte)

Löse das folgende lineare Gleichungssystem mit Hilfe der Gaußelimination mit kanonischer Pivotwahl und Rücksubstitution.

$$\begin{pmatrix} 2 & 4 & 0 \\ 1 & 3 & 2 \\ 2 & 3 & -1 \end{pmatrix} \cdot \mathbf{x} = \begin{pmatrix} 46 \\ 57 \\ 25 \end{pmatrix}$$

Antwort:

Aufgabe 3: (3 Punkte)

Invertiere die folgende Matrix mit Hilfe der Gausselimination mit kanonischer Pivotwahl.

$$\begin{pmatrix} 1 & 3 \\ 2 & 7 \end{pmatrix}$$

Antwort:

Aufgabe 4: (2 Punkte)

Was macht folgende Pythonfunktion? Was enthält `a` nach dem Ausführen des Algorithmus?

```
def doit(a):
    N, N = a.shape
    for k in range(0,N-1):
        for i in range(k+1,N):
            lam = a[i,k]/a[k,k]
            a[i,k+1:N] = a[i,k+1:N] - lam*a[k,k+1:N]
            a[i,k] = lam
    return a
```

Antwort:

2 Darstellung von Funktionen (10 Punkte)

Aufgabe 5: (2 Punkte)
Berechne das Taylorpolynom dritten Grades der Funktion $f(x) = \ln(x)$ um den Punkt $x_0 = 1$.

Antwort:

Aufgabe 6: (3 Punkte)
Berechne die Koeffizienten des Hornerschemas für ein Polynom 3. Grades mit den Nullstellen -1, 2, 5. Dabei sei der führende Koeffizient $c_3 = 1$.

Antwort:

Aufgabe 7: (2 Punkte)
Die Polynominterpolation mit Chebyshev-Stützstellen konvergiert garantiert. Warum benutzt man trotzdem manchmal die Interpolation an äquidistanten Stützstellen?

Antwort:

Aufgabe 8: (3 Punkte)
Berechne 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-Aitken-Schemas.

Antwort:

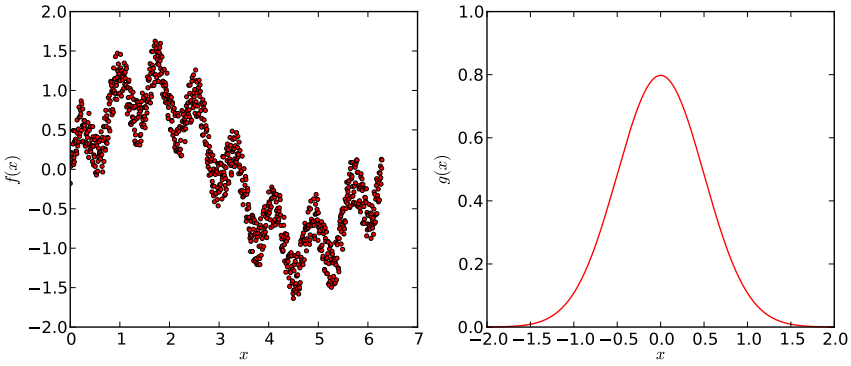
3 Signalverarbeitung (13 Punkte)

Aufgabe 9: (4 Punkte)
Schreibe eine Pythonfunktion `stats(x)`, die den Mittelwert und die Varianz einer Datenreihe berechnet und zurückgibt, ohne dabei die entsprechenden NumPy-Funktionen zu benutzen. Dabei sei x ein eindimensionales Numpy-Array.

Antwort:

Aufgabe 10: (3 Punkte)

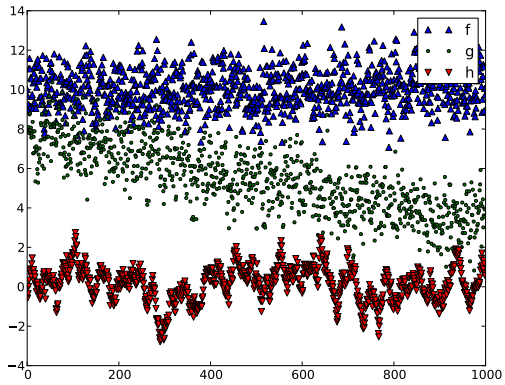
Skizziere die Faltung $f \star g$ der beiden in der folgenden Abbildung skizzierten Funktionen f (links) und g (rechts).

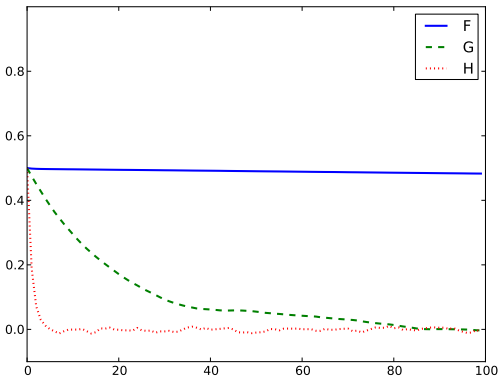


Antwort:

Aufgabe 11: (3 Punkte)

Ordne die in der linken Abbildung geplotteten Datenreihen f, g und h den in der rechten Abbildung geplotteten Autokorrelationsfunktionen F, G und H zu.





Antwort:

Aufgabe 12: (1 Punkt)

Das JPG-Format verwendet die *diskrete Kosinustransformation*, eine enge Verwandte der Fouriertransformation. Warum ist das Format gut geeignet zum Speichern von Fotos, aber nicht zum Speichern von mathematischen Plots?

Antwort:

Aufgabe 13: (2 Punkte)
In einer (equilibrierten) Computersimulation misst Du alle 200 Schritte den Druck P des Systems, und erhältst so $N = 50.000$ Messungen P_i, $i = 1(1)N$. Der mittlere Druck aller Messungen ist $\bar{P} = 2.0$, dessen Standardabweichung ist $\sigma = 2.0$, die integrierte Autokorrelationszeit ist $\tau_{\text{int}} = 100$ Messungen. Berechne den Fehler der Druckmessung (mit Konfidenzniveau 1σ).

Antwort:

4 Nichtlineare Gleichungssysteme (9 Punkte)

Aufgabe 14: (1 Punkt)
Warum gibt es keine Bisektionsmethode zur Berechnung einer Nullstelle einer Funktion in mehr als einer Dimension?

Antwort:

Aufgabe 15: (3 Punkte)

Bestimme die Nullstelle der Funktion $f(x) = x^2 - 3$ im Intervall $[0, 2]$ mit Hilfe des Bisektionsverfahrens und des Taschenrechners auf $\pm 0{,}1$.

Antwort:

Aufgabe 16: (5 Punkte)

Es sei eine Pythonfunktion `newton(f, fp, x0)` zur Nullstellensuche (dabei sei `f` die Funktion, `fp` deren Ableitung und `x0` der Startwert der Suche) und `exp(x)` zur Berechnung von e^x gegeben. Schreibe mit deren Hilfe eine Pythonfunktion `log(b,x)` die $\log_b(x)$ berechnet. Die Funktion `newton` muss hier *nicht* definiert werden, sondern wird als gegeben vorausgesetzt.

Antwort:

5 Numerisches Differenzieren und Integrieren (14 Punkte)

Aufgabe 17: (1 Punkt)

Die sogenannte Zustandssumme eines Zweiteilchensystems in einer Box sei

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

Welche Methode zur numerischen Integration würdest Du zur Berechnung dieses Integrals benutzen, und warum?

Antwort:

Aufgabe 18: (4 Punkte)

Schreibe eine Pythonfunktion `midpoint(f,a,b,N)`, die die zusammengesetzte Mittelpunktsregel implementiert und dadurch das Integral $\int_a^b f(x)\,\mathrm{d}x$ an N äquidistanten Stützstellen berechnet.

Antwort:

Aufgabe 19: (3 Punkte)

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

Antwort:

Aufgabe 20: (6 Punkte)

Wir betrachten einen getriebenen, gedämpften harmonischen Oszillator, der der Differentialgleichung

$$\ddot{x}(t) = -kx(t) - \gamma\dot{x}(t) + g(t)$$

unterliegt, wobei $g(x)$ periodisch mit Periodenlänge T sei. Wir wollen nun entsprechend die periodische Lösung dieser Differentialgleichung mit Hilfe eines finiten Differenzenschemas numerisch annähern. Benutze bei Schrittweite h für die zweite Ableitung die Dreipunktnäherung

$$\ddot{f}(t) \approx \frac{1}{h^2}\left[f(t-h) - 2f(t) + f(t+h)\right]$$

und für die Ableitung die zentrale Differenz

$$\dot{f}(t) \approx \frac{1}{2h}\left[f(t+h) - f(t-h)\right].$$

Beschreibe alle notwendigen Schritte, um eine Näherung für die Lösung der Differentialgleichung auf dem Intervall $[0, T[$ mit Schrittweite h und Startwert $x(0) = 0$ zu berechnen. Verfahren zur Lösung linearer Gleichungssysteme können vorausgesetzt werden.

Hinweis Beachte, dass Dein Gleichungssystem lösbar sein sollte, also quadratisch! Du musst also eine der Gleichungen, die Du aus der kanonischen Diskretisierung erhältst, durch die Randbedingung $x(0) = 0$ ersetzen.

Antwort:

6 Zufallszahlen (8 Punkte)

Aufgabe 21: (1 Punkt)
Nenne einen Vor- und einen Nachteil von Pseudozufallszahlen gegenüber "echten" Zufallszahlen.

Antwort:

Aufgabe 22: (2 Punkte)
Eine Methode zur Bestimmung der Qualität von Zufallszahlen ist, die Mittelwerte X_i von jeweils k Werten einer Reihe x_i von vermeintlichen Zufallszahlen zu bestimmen. Berechne die Standardabweichung $\sigma(X_i)$, die die Verteilung der Mittelwerte aufweisen sollte, damit die Reihe als Reihe von Standardzufallszahlen, also unabhängig und gleichverteilt in $[0, 1]$, gelten kann?

Antwort:

Aufgabe 23: (5 Punkte)
Schreibe eine Pythonfunktion `myrand()`, die eine Zufallszahl im Intervall $[0, 4]$ zurückliefert, die der Verteilung $P(x) = xe^{-x}$ genügt. Dabei soll sie die Pythonfunktion `random.random()` benutzen, die eine gleichverteilte Zufallszahl zwischen 0 und 1 erzeugt.

Antwort:

7 Lineare Algebra II (10 Punkte)

Aufgabe 24: (1 Punkt)

Welches numerische Verfahren kannst Du zum approximativen Lösen eines linearen Gleichungssystems verwenden? Nenne zwei Varianten.

Antwort:

Aufgabe 25: (1 Punkt)

Erstelle eine L+D+U-Zerlegung der Matrix

$$\begin{pmatrix} 0 & 3 & 1 & 7 \\ 1 & 0 & 4 & 1 \\ 4 & 1 & 3 & 0 \\ 3 & 5 & 1 & 1 \end{pmatrix}$$

Antwort:

Aufgabe 26: (4 Punkte)

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

```
from numpy import *
def Jacobi(A, b, x0, eps)
    N = len(x0)
    err = eps
    while (err >= eps):
        x = empty_like(x0)
        for i in range(N):
            x[i] = b[i]
            for j in range(N):
                if (j != i): x[i] += A[i,j]*x0[j]
            x /= A[j,j]
            err = amax(abs(x0-x))
            x0 = x
       return x
```

Antwort:

Aufgabe 27: (4 Punkte)

Wende die Gram-Schmid-Methode zur Orthogonalisierung der folgenden Matrix an.

$$A = \begin{pmatrix} 1 & 3 & 6 \\ -2 & 4 & -7 \\ -2 & 5 & 1 \end{pmatrix}$$

Antwort:

8 Optimierung (7 Punkte)

Aufgabe 28: (1 Punkt)
Was ist der Sinn einer Schrittweitensteuerung bei lokalen Minimierungsalgorithmen?

Antwort:

Aufgabe 29: (3 Punkte)
Gesucht ist ein lokales Minimum der Funktion $f(x,y) = x^2 + y^2 + (x-2)^2 + (y-1)^2$. Führe (mit Hilfe des Taschenrechners) vom Ausgangspunkt $x = y = 0$ drei Schritte des Verfahrens des steilsten Abstiegs mit Schrittweite $\lambda = 0{,}2$ aus.

Antwort:

Aufgabe 30: (3 Punkte)
Bringe das Problem

$$\min_{x,y\in\mathbb{R}^2} (\pi, 1)^T (x, y) \quad \text{unter den Nebenbedingungen} \quad y \geq 0,\ y \leq x \text{ und } y \leq 1 - x$$

auf Simplex-Normalform. Begründe Deine Umformungen!

Antwort:

9 Differentialgleichungen (12 Punkte)

Aufgabe 31: (1 Punkt)
Was ist der Unterschied in der Implementierung zwischen einer impliziten und einer expliziten Runge-Kutta-Methode?

Antwort:

Aufgabe 32: (1 Punkt)

Es sei die Differentialgleichung erster Ordnung

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

gegeben. Formuliere diese Differentialgleichung als Gleichung zweiter Ordnung.

Antwort:

Aufgabe 33: (4 Punkte)

Gegeben sei folgendes Butcher-Tableau für die *Heun*-Methode:

$$\begin{array}{c|cc} 0 & & \\ 1 & 1 & \\ \hline & 1/2 & 1/2 \end{array}$$

Führe mit Hilfe des Taschenrechners zwei Schritte der Methode für die Differentialgleichung $\dot{y} = f(t, y) = y\,\mathrm{e}^t$ mit Schrittweite $h = 0{,}1$ und Anfangsbedingung $y(t = 0) = 1$ durch.

Hinweis Die analytische Lösung der DGL ist $y(t) = \exp(\mathrm{e}^t - 1)$.

Antwort:

Aufgabe 34: (6 Punkte)

Leite die sogenannten Verlet-Methode zum numerischen Annähern der Bewegungsgleichung her. Entwickle dazu die Position $r(t)$ an der Stelle t und werte das Taylorpolynom an den Stellen $t + h$ und $t - h$ bis zur vierten Ordnung aus. Gib daraus eine Gleichung zur numerischen Berechung der Position zum Zeitpunkt $t + h$ an. Worin unterscheidet sich diese Methode grundlegend von einer Runge-Kutta-Methode? Beschreibe, wie mit Hilfe der Gleichung die Bewegungsgleichung angenähert werden kann. Welche Fehlerordnung hat der Algorithmus?

Antwort:

10 Programmieren (7 Punkte)

Aufgabe 35: (6 Punkte)

Schreibe zwei Versionen einer Pythonfunktion, die die Summe $\sum_{i=0}^{N-k-1} x_i x_{i+k}$ einer Datenreihe x_i berechnet, die im NumPy-Array `x` gespeichert ist. `sum_python(x)` soll dabei eine Schleife in Python verwenden, `sum_numpy(x)` einen einzigen NumPy-Ausdruck.

Antwort:

Aufgabe 36: (1 Punkt)

Wenn Du zwei Pythonfunktionen hast, die beide dasselbe berechnen, aber einmal mit Hilfe einer Schleife in Python und einmal in einem NumPy-Ausdruck, welche der beiden Funktionen ist typischerweise schneller, und warum?

Antwort: