Universität Stuttgart – Institut für Computerphysik

Klausur

Physik auf dem Computer SS 2012

JP Dr. Axel Arnold   Dr. Olaf Lenz   Florian Fahrenberger
Dominic Röhm

15. August 2012

Name	
Vorname	
Matrikelnummer	

Hinweise

- In der Regel gibt der verfügbare freie Platz einen Hinweis darauf, welchen Umfang die Lösung haben sollte.
- 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 (7 Punkte)

Aufgabe 1: (1 Punkt)

Nenne einen Vorteil der Spaltenpivotwahl gegenüber der kanonischen Pivotwahl bei der Gaußeliminiation einer quadratischen Matrix?

Antwort:

Im Gegensatz zur kanonischen Pivotisierung führt die Gaußeliminiation mit Spaltenpivotwahl bei quadratischen Matrizen immer zum Ergebnis, wenn die Matrix regulär ist.

Aufgabe 2: (5 Punkte)

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

$$\begin{pmatrix} 16 & 12 & 6 \\ -8 & -2 & 0 \\ 32 & 16 & 8 \end{pmatrix} \cdot \mathbf{x} = \begin{pmatrix} 408 \\ -136 \\ 704 \end{pmatrix}$$

Antwort:

$$\left(\begin{array}{ccc|c} 16 & 12 & 6 & 408 \\ -8 & -2 & 0 & -136 \\ \mathbf{32} & 16 & 8 & 704 \end{array}\right) \quad \text{(Tausche Zeile 1 und 3)}$$

$$\Rightarrow \left(\begin{array}{ccc|c} 32 & 16 & 8 & 704 \\ -8 & -2 & 0 & -136 \\ 16 & 12 & 6 & 408 \end{array}\right) \quad \text{(Eliminiere Zeilen 2 und 3))}$$

$$\Rightarrow \left(\begin{array}{ccc|c} 32 & 16 & 8 & 704 \\ 0 & 2 & 2 & 40 \\ 0 & 4 & 2 & 56 \end{array}\right) \quad \text{(Tausche Zeilen 2 und 3)}$$

$$\Rightarrow \left(\begin{array}{ccc|c} 32 & 16 & 8 & 704 \\ 0 & 4 & 2 & 56 \\ 0 & 2 & 2 & 40 \end{array}\right) \quad \text{(Eliminiere Zeile 3)}$$

$$\Rightarrow \left(\begin{array}{ccc|c} 32 & 16 & 8 & 704 \\ 0 & 4 & 2 & 56 \\ 0 & 0 & 1 & 12 \end{array}\right) \quad \text{(Rücksubstitution)}$$

$$\Rightarrow \quad \mathbf{x} = \begin{pmatrix} 15 \\ 8 \\ 12 \end{pmatrix}$$

Aufgabe 3: (1 Punkt)
Erstelle eine reguläre (3×3)-Matrix, bei der es unmöglich ist, sie mit Hilfe der Gaußelimination mit kanonischer Pivotisierung zu diagonalisieren.

Antwort:

$$\begin{pmatrix} 0 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{pmatrix} \quad \text{oder} \quad \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

Die 0 auf der Diagonalen verhindert die Gaußelimination, obwohl die Matrix regulär ist.

2 Darstellung von Funktionen (14 Punkte)

Aufgabe 4: (1 Punkt)
Worin unterscheidet sich die Hermite-Interpolation von der Lagrange-Interpolation?

Antwort:

Bei der Hermite-Interpolation verwendet man an den Stützstellen zusätzlich zu den Funktionswerten (Lagrange-Interpolation) auch noch Ableitungen. In gewisser Weise handelt es sich um mehrfache Stützpunkte.

Aufgabe 5: (1 Punkt)
Welchen Vorteil hat die Polynominterpolation mit Chebyshev-Stützstellen gegenüber der Polynominterpolation mit äquidistanten Stützstellen?

Antwort:

Für höhere Anzahl von Stützstellen wird die Interpolation mit Chebyshev-Stützstellen immer genauer, während man bei der Interpolation mit äquidistanten Stützstellen davon nicht ausgehen kann.

Aufgabe 6: (5 Punkte)

Berechne mit Hilfe der Lagrangepolynome das interpolierende Polynom an den Stützstellen $-\pi, 0, \pi$ zur Funktion $f(x) = \cos x$.

Antwort:

$\mathbf{k}$	$\mathbf{x_k}$	$\mathbf{y_k}$
0	$-\pi$	-1
1	0	1
2	π	-1

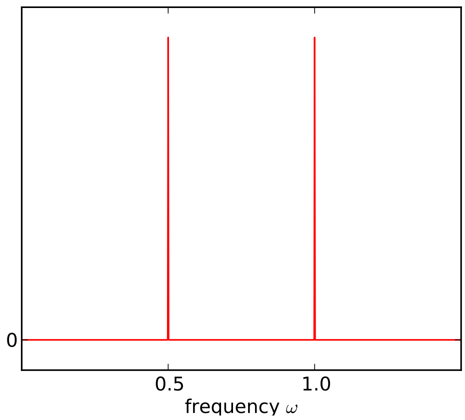
$$\begin{aligned} P(x) &= \sum_{k=0}^{2} y_k L_k(x) \\ &= y_0 \left(\frac{x^2 - \pi x}{2\pi^2} \right) + y_1 \left(\frac{x^2 - \pi^2}{-\pi^2} \right) + y_2 \left(\frac{x^2 + \pi x}{2\pi^2} \right) \\ &= - \left(\frac{x^2 - \pi x}{2\pi^2} \right) + \left(\frac{x^2 - \pi^2}{-\pi^2} \right) - \left(\frac{x^2 + \pi x}{2\pi^2} \right) \end{aligned}$$

$$P(x) = -\frac{2x^2}{\pi^2} + 1$$

Aufgabe 7: (3 Punkte)

Skizziere graphisch die Fouriertransformierte der Funktion $f(x) = \sin \pi x + \sin 2\pi x$. Bitte denke daran, die Achsen zu beschriften!

Antwort:



Aufgabe 8: (1 Punkt)

Nenne einen Vorteil der Splineinterpolation gegenüber der Interpolation mit Polynomen.

Antwort:

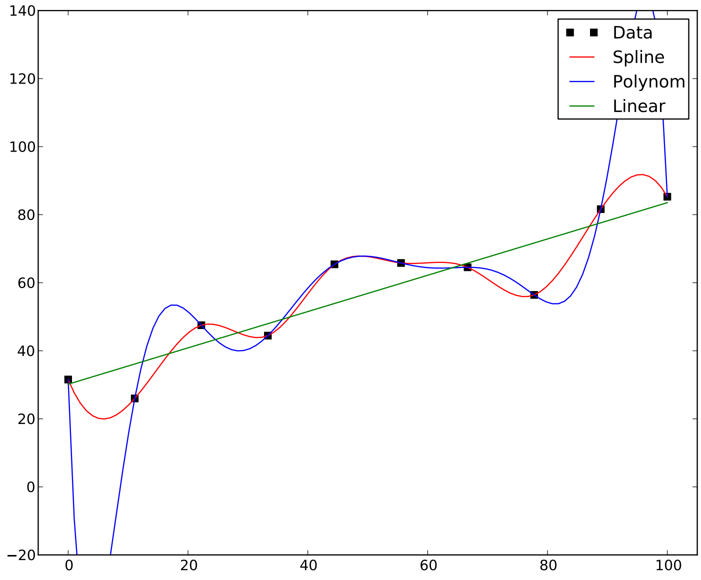
- kein Rungephänomen
- lokal zu berechnen
- glattere Funktion
- Randbedingungen wählbar

Aufgabe 9: (3 Punkte)

Im folgenden Graph sind n Messpunkte eingezeichnet. Skizziere die Graphen der Funktionen, die bei den folgenden Methoden entstehen:

- Polynominterpolation $(n-1)$-ten Grades
- Natürliche Splines
- Methode der kleinsten Quadrate mit linearer Fitfunktion

Antwort:



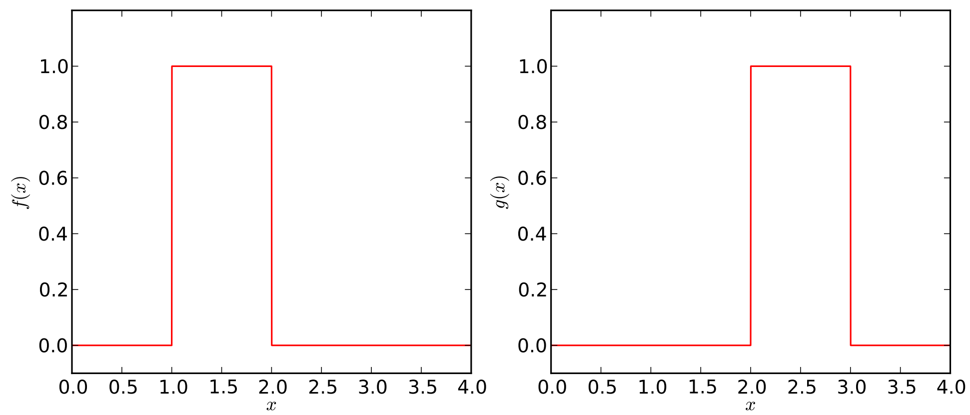
3 Signalverarbeitung (11 Punkte)

Aufgabe 10: (2 Punkte)
Nenne einen Vor- und einen Nachteil der sogenannten „Schnelle Fouriertransformation“ (FFT) gegenüber der diskreten Fouriertransformation (DFT).

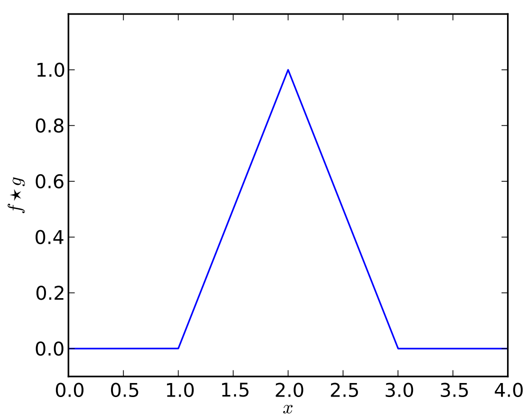
Antwort:

- Vorteil: Sie ist schneller.
- Nachteil: Sie erfordert als Anzahl von Stützstellen eine Zweierpotenz.

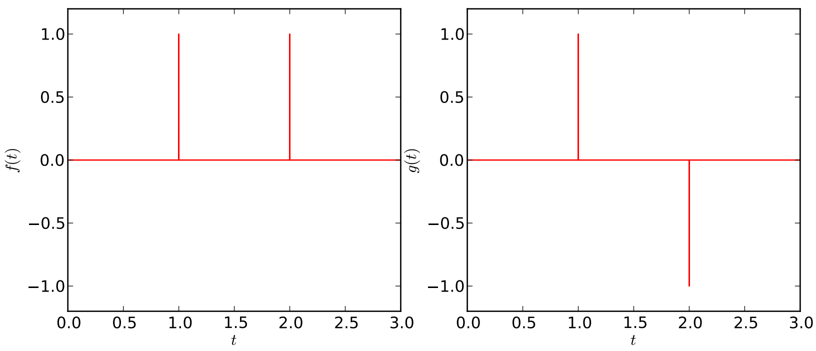
Aufgabe 11: (3 Punkte)
Skizziere das Faltungsprodukt $f \star g$ der beiden in der folgenden Abbildung skizzierten Funktionen f (links) und g (rechts).



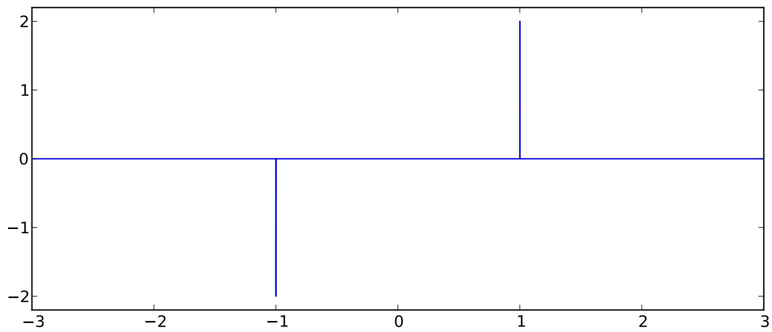
Antwort:



Aufgabe 12: (3 Punkte)
Skizziere die Kreuzkorrelation der beiden in der folgenden Abbildung skizzierten Funktionen f (links) und g (rechts).



Antwort:



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

Antwort:

Wenn man ein Signal mit einer gegebenen Rate abtastet, dann gibt die Nyquist-Frequenz die maximale Frequenz an, die im Signal noch aufgelöst werden kann.

4 Nichtlineare Gleichungssysteme (14 Punkte)

Aufgabe 14: (1 Punkt)
Welche der folgenden drei Methoden kannst Du anwenden, wenn Du die Nullstelle einer Funktion suchst, deren Ableitung Du nicht kennst?

- Newtonverfahren
- Bisektion
- Regula falsi

Antwort:

Bisektion und Regula falsi.

Aufgabe 15: (5 Punkte)
Schreibe ein Stück Pythoncode, das mit Hilfe der (gegebenen) Funktion `newton(f, fp, x0)` zur Nullstellensuche den Wert von $17^{\frac{1}{4}}$ berechnet und ausgibt, ohne dabei den Pythonoperator `**` zu verwenden! Verwende als Startwert $x_0 = 2$. `f` sei dabei eine mathematische Funktion, `fp` deren Ableitung, und `x0` der Startwert des Newtonverfahrens. Die Funktion `newton` muss hier *nicht* definiert werden, sondern wird als gegeben vorausgesetzt.

Antwort:

```
def f(x):
  return (x*x*x*x - 17)

def fp(x):
  return 4.0*x*x*x

print newton(f, fp, 2.0)
```

Aufgabe 16: (5 Punkte)

Berechne die ersten drei Schritte des Sekantenverfahrens zum Lösen von $x^2 - 2 = 0$ mit den Startwerten $a = 0$ und $b = 2$.

Antwort:

Sekantenverfahren: $x_{n+1} = x_n - f(x_n)\frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}$

$$
\begin{aligned}
x_0 &= 0 \\
x_1 &= 2 \\
x_2 &= 2 - f(2)\frac{2-0}{f(2)-f(0)} = 2 - 2\frac{2}{4} = 1 \\
x_3 &= 1 - f(1)\frac{1-2}{f(1)-f(2)} = 1 - (-1)\frac{-1}{-3} = \frac{4}{3} \\
x_4 &= \frac{4}{3} - f(\frac{4}{3})\frac{\frac{4}{3}-1}{f(\frac{4}{3})-f(1)} = \frac{4}{3} - (-\frac{2}{9})\frac{\frac{1}{3}}{(-\frac{2}{9})-(-1)} = \frac{10}{7} = 1.428\ldots
\end{aligned}
$$

Aufgabe 17: (3 Punkte)

Konvergiert die Regula falsi schneller als das Sekantenverfahren? Warum?

Antwort:

Nein. Die Regula falsi macht entweder einen Bisektionsschritt und konvergiert damit so langsam wie die Bisektion, oder selbst einen Sekantenschritt.

5 Numerisches Differenzieren und Integrieren (9 Punkte)

Aufgabe 18: (2 Punkte)

Welchen Vorteil haben Quasizufallszahlen bei der Monte-Carlo-Integration gegenüber Pseudozufallszahlen, und warum?

Antwort:

Mit Quasizufallszahlen konvergiert das Ergebnis einer Monte-Carlo-Integration schneller, weil die Quasizufallszahlen gleichmäßiger im Intervall verteilt sind, und damit die *Diskrepanz* der Zufallsfolge minimieren, die für die Konvergenz notwendig ist.

Aufgabe 19: (3 Punkte)

Diskretisiere die Differentialgleichung $\frac{\mathrm{d}^2 f(x)}{\mathrm{d}x^2} + \alpha x = g(x)$ mit Hilfe eines finite Differenzen-Schemas. Benutze für die zweite Ableitung die folgende Näherung:

$$f''(x_i) \approx \frac{1}{h^2}(f(x_{i-1}) - 2f(x_i) + f(x_{i+1})).$$

Beschreibe alle notwendigen Schritte, um eine Näherung für die Lösung der Differentialgleichung auf dem Interval $[0, L]$ zu berechnen. Verfahren zur Lösung gewöhnlicher Gleichungssysteme können vorausgesetzt werden.

Antwort:

Die Randbedingungen können nach Belieben gewählt werden, solange das GLS lösbar bleibt.

1. Erzeuge N äquidistante Stützstellen x_i im Intervall $[0,L]$.
2. Löse das folgende Gleichungssystem (z.B. mittels Gausselimination), um die Näherung für die Lösung der DGL f_i an den Stützstellen x_i zu erhalten:

$$\begin{pmatrix} -\frac{2}{h^2} & \frac{1}{h^2} & 0 & 0 & \cdots & 0 \\ \frac{1}{h^2} & -\frac{2}{h^2} & \frac{1}{h^2} & 0 & \ddots & 0 \\ 0 & \frac{1}{h^2} & -\frac{2}{h^2} & \frac{1}{h^2} & \ddots & 0 \\ 0 & 0 & \frac{1}{h^2} & -\frac{2}{h^2} & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \frac{1}{h^2} \\ 0 & 0 & 0 & 0 & \frac{1}{h^2} & -\frac{2}{h^2} \end{pmatrix} \cdot \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_{n-1} \\ f_n \end{pmatrix} = \begin{pmatrix} g(x_1) - \alpha x_1 \\ g(x_2) - \alpha x_2 \\ \vdots \\ g(x_{n-1}) - \alpha x_{n-1} \\ g(x_n) - \alpha x_n \end{pmatrix}$$

Aufgabe 20: (4 Punkte)
Schreibe eine Pythonfunktion `trapez(f,a,b,N)`, die die zusammengesetzte Trapezregel implementiert und dadurch das Integral $\int_a^b f(x)\,\mathrm{d}x$ an N äquidistanten Stützstellen berechnet. Dabei seien a und b selbst Stützstellen.

Antwort:

```
def trapez(f, a, b, N=100):
    h = abs( (b-a)/(N-1) )
    integral = 0.5 * (f(a)+f(b))
    for i in range(1, N-1):
        integral += f(a+i*h)
    return integral*h
```

6 Zufallszahlen (10 Punkte)

Aufgabe 21: (1 Punkt)
Nenne drei Methoden, um die Qualität von Zufallszahlen zu überprüfen.

Antwort:

- Histogramm der Verteilung
- χ^2-Test
- Fourieranalyse

Aufgabe 22: (4 Punkte)
Skizziere (graphisch oder verbal), wie man mit Hilfe der Verwerfungsmethode aus einer Folge von gleichverteilten Zufallszahlen eine Folge von Zufallszahlen erzeugen kann, die eine durch die Verteilungsfunktion $G(x)$ gegebene Verteilung hat.

Antwort:

Sei $[a,b]$ ein Intervall, dass den Träger von G umfasst, und $G_0 > G(x)$ für alle $x \in [a,b]$. Dann ziehe eine Zufallszahl $p \in [a,b]$ gleichverteilt und eine zweite $u \in [0,G_0]$, ebenfalls gleichverteilt. Ist $u < G(p)$, so benutze p als nächstes Element der Zufallsfolge, andernfalls fange vorne mit der Ziehung von p und u an.

Aufgabe 23: (0 Punkte)

Skizziere (graphisch, verbal oder audiovisuell) ob es Poincaré-Schnitten auch mit Himbeerfüllung gibt.

Antwort:

Vielleicht.

Aufgabe 24: (1 Punkt)

Welchen Nachteil hat die Verwerfungsmethode gegenüber der Inversionsmethode?

Antwort:

- Bei der Verwerfungsmethode werden viele Zufallszahlen verworfen.
- Die Verwerfungsmethode funktioniert nur für beschränkte Verteilungen mit kompaktem Träger.

Aufgabe 25: (4 Punkte)

Angenommen, Du möchtest die Landoberfläche der Erde mit Hilfe der Monte-Carlo-Integration bestimmen. Dazu erzeugst Du zufällige Positionen auf der Erdoberfläche, indem Du jeweils den Breiten- und den Längengrad durch gleichverteilte Zufallszahlen festlegst. Wieso wirst Du keine gute Näherung erhalten?

Antwort:

Die Positionen auf der Oberfläche sind nicht gleichverteilt, deswegen wird die Näherung nicht gut sein. Je näher ein Flächenelement am Pol liegt, desto größer wird sein Gewicht.

7 Lineare Algebra II (9 Punkte)

Aufgabe 26: (2 Punkte)

Wird für die folgende Matrix das Jacobi-Verfahren konvergieren? Warum?

$$\begin{pmatrix} 6 & 3 & 1 & -2 \\ -5 & 8 & 2 & 3 \\ 2 & -1 & 3 & 0 \\ 5 & -3 & 1 & 10 \end{pmatrix}$$

Antwort:

Nein, da die Matrix nicht diagonaldominant ist:

$$8 = |a_{22}| = |a_{21}| + |a_{23}| + |a_{24}| = |-5| + |2| + |3|$$

Aufgabe 27: (1 Punkt)

Zu welchem Verfahren ist das Relaxationsverfahren mit einem Dämpfungsparameter von $\omega = 1$ äquivalent?

Antwort:

Gauß-Seidel-Verfahren

Aufgabe 28: (3 Punkte)

Was tut die folgende Pythonfunktion? Welcher Algorithmus ist dabei implementiert?

```
def f(A, N=1000):
    n = A.shape[0]
    I = numpy.identity(n)
    Ak = A.copy()
    for i in range(N):
        shift = Ak[-1,-1]
        Q, R = scipy.linalg.qr(Ak - shift*I)
        Ak = numpy.dot(R, Q) + shift*I
    lambdas = numpy.array([Ak[i,i] for i in range(n)])
    return lambdas
```

Antwort:

Sie berechnet die Eigenwerte der Matrix `A` mit Hilfe des QR-Algorithmus.

Aufgabe 29: (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:

$$L = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 4 & 1 & 0 & 0 \\ 3 & 5 & 1 & 0 \end{pmatrix} D = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} U = \begin{pmatrix} 0 & 3 & 1 & 7 \\ 0 & 0 & 4 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Aufgabe 30: (2 Punkte)

Ist das Gauß-Seidel-Verfahren gut für parallele Ausführung geeignet? Warum?

Antwort:

Nein, da jeder einzelne Rechenschritt vom Ergebnis des vorherigen abhängt. Allerdings gibt es Varianten wie Red-Black-Gauß-Seidel, die für bestimmte Stencils, etwa den für Laplace-Stencil, sehr wohl gut parallelisieren.

8 Optimierung (7 Punkte)

Aufgabe 31: (4 Punkte)

Gesucht ist das lokale Minimum der Funktion $f(x, y) = \cos((x+1)(y-1))$. Führe vom Ausgangspunkt $x = y = 0$ zwei Schritte des Verfahrens des steilsten Abstiegs mit Schrittweite $\lambda = 0{,}1$ aus.

Antwort:

$$
\begin{aligned}
\nabla f(\vec{x}) &= \begin{pmatrix} -(y-1)\sin((x+1)(y-1)) \\ -(x+1)\sin((x+1)(y-1)) \end{pmatrix} \\
\vec{x}_{i+1} &= \vec{x}_i - \lambda \nabla f(\vec{x}_i) \\
\vec{x}_0 &= \begin{pmatrix} 0 \\ 0 \end{pmatrix} \\
\vec{x}_1 &= \vec{x}_0 - \lambda \begin{pmatrix} -0.8415 \\ 0.8415 \end{pmatrix} = \begin{pmatrix} 0.08415 \\ -0.08415 \end{pmatrix} \\
\vec{x}_2 &= \begin{pmatrix} 0.08415 \\ -0.08415 \end{pmatrix} - \lambda \begin{pmatrix} -1.00049 \\ 1.00049 \end{pmatrix} = \begin{pmatrix} 0.18415 \\ -0.18415 \end{pmatrix}
\end{aligned}
$$

Aufgabe 32: (3 Punkte)

Skizziere (verbal oder graphisch) die Idee des *Simulated Annealing*.

Antwort:

Wir betrachten eine zu minimierende Funktion E als Energielandschaft. Bei hohen Temperaturen wird eine gewöhnliche Metropolissimulation alle Zustände etwa gleich häufig besuchen, da die Wahrscheinlichkeit $\sim e^{-\beta E(x)}$ ist. Wenn man nun langsam abkühlt, als $\beta \to \infty$ wächst, werden die niedrigen Energien immer wahrscheinlicher, so dass letztlich die nur noch das globale Minimum wahrscheinlich ist. Eine Monte-Carlo-Simulation sollte dann dorthin konvergieren.

9 Programmieren (8 Punkte)

Aufgabe 33: (5 Punkte)
Schreibe eine Pythonfunktion `random()`, die einen linearen Kongruenzgenerator nach der Formel $x_{i+1} = (ax_i + b) \bmod m$ für $a = 1103515245$, $m = 2^{31}$ und $b = 12345$ implementiert.

Antwort:

```
def random():
    global state
    state = (1103515245*state + 12345) % 2**31
    return state
```

Aufgabe 34: (3 Punkte)
Forme die folgende Pythonschleife in einen entsprechenden NumPy-Ausdruck (ohne Schleife in Python) um.

```
for i in range(N+1): a[i] += b[N-i]
```

Antwort:

```
a[:N+1] += b[N::-1]
```

10 Weitere Aufgaben (11 Punkte)

Aufgabe 35: (6 Punkte)

Berechne die Lösung der Gleichung $x^2 = e^x$ numerisch mit Hilfe eines Taschenrechners und einer der Methoden aus der Vorlesung auf zwei Nachkommastellen genau. Wie gehst Du vor? Welche Methode verwendest Du?

Antwort:

1. Transformation der Gleichung $x^2 = e^x \Rightarrow e^x - x^2 = 0$
2. Nullstellensuche mit Hilfe des Newtonverfahrens:
   - Funktion: $f(x) = e^x - x^2$
   - Ableitung: $f'(x) = e^x - 2x$
   - Newtonverfahren: $x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$
   - Rechnung:

$$
\begin{array}{lll}
 & & x_0 = 0 \\
f(x_0) = 1 & f'(x_0) = 1 & x_1 = -1 \\
f(x_1) = -0{,}632 & f'(x_1) = 2{,}368 & x_2 = -0{,}733 \\
f(x_2) = -0{,}057 & f'(x_2) = 1{,}947 & x_3 = -0{,}704 \\
f(x_3) = -0{,}001 & f'(x_3) = 1{,}902 & x_4 = -0{,}703
\end{array}
$$

   - Ergebnis: $-0{,}70$

Aufgabe 36: (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? Skizziere die Graphen der mittleren Abweichungen vom Ursprung über der Anzahl Schritte.

Antwort:

Es gilt für die Verteilung nach t Schritten

$$P(x) \approx \frac{1}{\sqrt{2\pi t}\sigma} e^{-\frac{1}{2\sigma^2 t}x^2}$$

und damit

$$\langle |x| \rangle = \int |x| P(x) dx \approx 2 \int_0^\infty x \frac{1}{\sqrt{2\pi t}\sigma} e^{-\frac{1}{2\sigma^2 t}x^2} dx = \sqrt{t}\sigma \frac{2}{\sqrt{2\pi}}$$

während

$$\langle x^2 \rangle = \int x^2 P(x) dx \approx \int_{-\infty}^\infty x^2 \frac{1}{\sqrt{2\pi t}\sigma} e^{-\frac{1}{2\sigma^2 t}x^2} dx = t\sigma^2$$

bzw. $\sqrt{\langle x^2 \rangle} \approx \sqrt{t}\sigma$.
Für die andere Komponente gilt der Erwartungswert $\langle |y| \rangle = \langle y \rangle = t/4$,
also 50 bzw 200.