Prof. Dr. Ulrich Rüde
Lehrstuhl für Systemsimulation
FAU Erlangen-Nürnberg

Algorithmik kontinuierlicher Systeme — 12. Februar 2019

Angaben zur Person (Bitte in DRUCKSCHRIFT ausfüllen!):

Name, Vorname: ____________________

Geburtsdatum: ____________________

Matrikelnummer: ____________________

Studienfach: ____________________

Nicht von der Kandidatin bzw. vom Kandidaten auszufüllen!

Aufgabe	1	2	3	4	5	6	7	8	9
Max. Punktzahl	10	9	8	13	9	10	14	10	7
Erreichte Punkte									

Gesamtpunktzahl	
Note	

Organisatorische Hinweise

Die folgenden Hinweise bitte aufmerksam lesen und die Kenntnisnahme durch Unterschrift bestätigen!

- Bitte legen Sie Ihren Studentenausweis und einen Lichtbildausweis zur Personenkontrolle bereit.
- Hilfsmittel (außer Schreibmaterial und Lineal) sind nicht zugelassen. Elektronische Geräte sind auszuschalten.
- Fragen zu den Prüfungsaufgaben werden grundsätzlich nicht beantwortet.
- Die Lösung einer Aufgabe muss auf das jeweilige Aufgabenblatt geschrieben werden. Sollte der Platz nicht reichen, verwenden Sie die Zusatz-Seiten am Ende der Klausur. Fügen Sie einen Hinweis in Ihre Lösung ein, wenn die Lösung auf den Zusatz-Seiten fortgesetzt wurde und beschriften Sie diese mit Namen und Aufgabennummer.
- Es können durch die Aufsicht zusätzlich Seiten eingeheftet werden, sollte mehr Platz benötigt werden. Bitte beschriften Sie den Kopf dieser Seiten mit Ihrem Namen und Matrikelnummer. Streichen Sie alles, was nicht verwendet werden soll, doppelt aus.
- Die Programmieraufgaben sind in der Programmiersprache Python 3 zu bearbeiten.
- Wenn Sie die Prüfung aus gesundheitlichen Gründen abbrechen müssen, so muss Ihre Prüfungsunfähigkeit durch eine Untersuchung bei einem Vertrauensarzt nachgewiesen werden. Melden Sie sich bei der Aufsicht und lassen Sie sich das entsprechende Formular aushändigen.
- Die Bearbeitungszeit beträgt 90 Minuten.
- Überprüfen Sie die Prüfungsaufgaben auf Vollständigkeit (20 Seiten inklusive Deckblatt) und einwandfreies Druckbild.
- Vergessen Sie nicht, auf dem Deckblatt die Angaben zur Person einzutragen und die **Erklärungen auf dieser Seite zu unterschreiben**.
- Viel Erfolg!

Erklärungen

Durch meine Unterschrift bestätige ich den Empfang der vollständigen Klausurunterlagen und die Kenntnisnahme der obigen Informationen.

Erlangen, 12. Februar 2019 ______________________________

(Unterschrift)

Aufgabe 1 — Theoriefragen (10 Punkte)

a) Beantworten Sie die folgenden Fragen! Schreiben Sie Ihre Antwort in die rechte Spalte der Tabelle!

Sei $\mathbf{A}$ eine untere Dreiecksmatrix mit Dimension $(n \times n)$. Welche Komplexität hat die Vorwärts-Substitution für $\mathbf{A}$.	$\mathcal{O}(\qquad)$
Ein Zelle im D2Q9 Lattice-Boltzmann Verfahren hat die Einträge $f_1, \ldots, f_9$. Wie berechnen Sie daraus die Geschwindigkeit $\vec{u}$ dieser Zelle?	$\vec{u} =$
Sei $\mathbf{A}$ eine dünnbesetzte $(n \times n)$-Matrix mit höchstens neun von Null verschiedenen Einträgen pro Zeile. Welche Komplexität hat die Berechnung von zwei Iterationsschritten des Gauß-Seidel-Verfahrens zur Lösung von $\mathbf{Ax} = \mathbf{b}$.	$\mathcal{O}(\qquad)$
Sei $\mathbf{D}$ eine invertiarbare $(n \times n)$-Diagonalmatrix. Welche Komplexität hat die Berechnung der Inversen von $\mathbf{D}$.	$\mathcal{O}(\qquad)$
Welche Konditionszahl besitzt eine Rotationsmatrix bezüglich der Spektralnorm?	
Wie groß ist der Approximationsfehler des B-Spline-Interpolanten bei Schrittweite h?	$\mathcal{O}(\qquad)$

b) Sind folgende Gleichungssysteme $Ax = b$ überbestimmt, unterbestimmt oder eindeutig lösbar?

	überbestimmt	unterbestimmt	eindeutig lösbar
$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 3 \\ 3 \\ 1 \end{bmatrix}$	☐	☐	☐
$\begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 6 \\ 3 & 6 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 3 \\ 6 \\ 6 \end{bmatrix}$	☐	☐	☐
$\begin{bmatrix} 2 & 2 \\ 3 & 9 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$	☐	☐	☐
$\begin{bmatrix} 1 & 4 & 3 \\ 0 & 1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 7 \\ 8 \end{bmatrix}$	☐	☐	☐

Aufgabe 2 — QR-Zerlegung (9 Punkte)

Von der Matrix A ist folgende QR-Zerlegung bekannt:

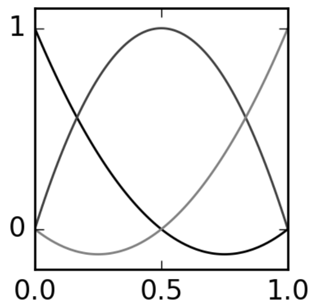
$$A = \begin{bmatrix} -2 & 0 & 0 & 2 \\ 2 & 2 & 6 & 4 \\ 2 & 2 & 10 & -2 \\ 2 & 0 & -4 & -4 \end{bmatrix} = \underbrace{\begin{bmatrix} -\frac{1}{2} & \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & \frac{1}{2} \end{bmatrix}}_{Q} \underbrace{\begin{bmatrix} 4 & 2 & 6 & -2 \\ 0 & 2 & 10 & 4 \\ 0 & 0 & 4 & -2 \\ 0 & 0 & 0 & -4 \end{bmatrix}}_{R}$$

a) Lösen Sie das lineare Gleichungssystem $Ax = b$ für $b = [-2, 0, 2, 4]^T$ nach x

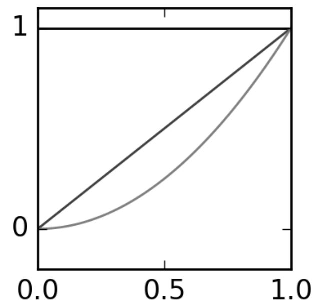
b) berechnen Sie den Betrag der Determinante von A.

Aufgabe 3 — Interpolation (8 Punkte)

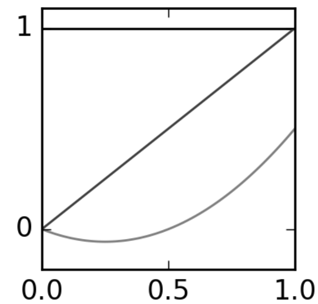
a) Jedes dieser Bilder beschreibt eine Basis für Polynome zweiten Grades mit den Stützstellen 0, $\frac{1}{2}$ und 1. Schreiben Sie unter jedes Bild, um welche Art von Basis es sich handelt (der Name genügt).



Typ:



Typ:



Typ:

b) Bestimmen Sie die Koeffizienten des Newton-Interpolationspolynoms zu den Folgenden Punkten:

i	0	1	2
x_i	0	2	4
y_i	1	2	-1

Aufgabe 4 — Python (13 Punkte)

a) Lösen Sie die folgenden Probleme mit einer Funktion in Python!
`numpy` ist als `np` importiert.

```
import numpy as np
```

- **Beispiel**: Prüfen, ob die Ganzzahl `n` gerade ist.

```
def ganzzahl(n):
    return n % 2 == 0
```

- Die Liste `L` in umgekehrter Reihenfolge zurückgeben.

- Die Summe aller Elemente einer beliebigen Liste `L` zurück geben.

- Zählen wie oft in zwei gleich langen Listen a und b gilt: $a_i! = b_i$.

- Die Länge der Liste `l1` oder `l2`, je nachdem welche kürzer ist.

b) Was geben folgende Codeschnipsel aus?

```
a = -1
for i in range(10):
    if i % 2 == 0:
        a += 1
print(a)
```

Ausgabe: _

```
A = np.ones((2, 2))
A[:, 1] = 2
A[1, :] = list(range(5, 7))
print((A[0, :] + A[1, :])[1])
```

Ausgabe: _

```
def magic(l1, l2):
    summe = sum(l1) + sum(l2)
    if l2[0] == 5:
        return l2[0]
    else:
        return l1[-1]


A = np.array([0, 5, 2])
print(magic(A, A[1:]))
```

Ausgabe: _

```
a = [1, 2, 3, 4]
a = list(map(lambda x: x**2, a))
print(a)
```

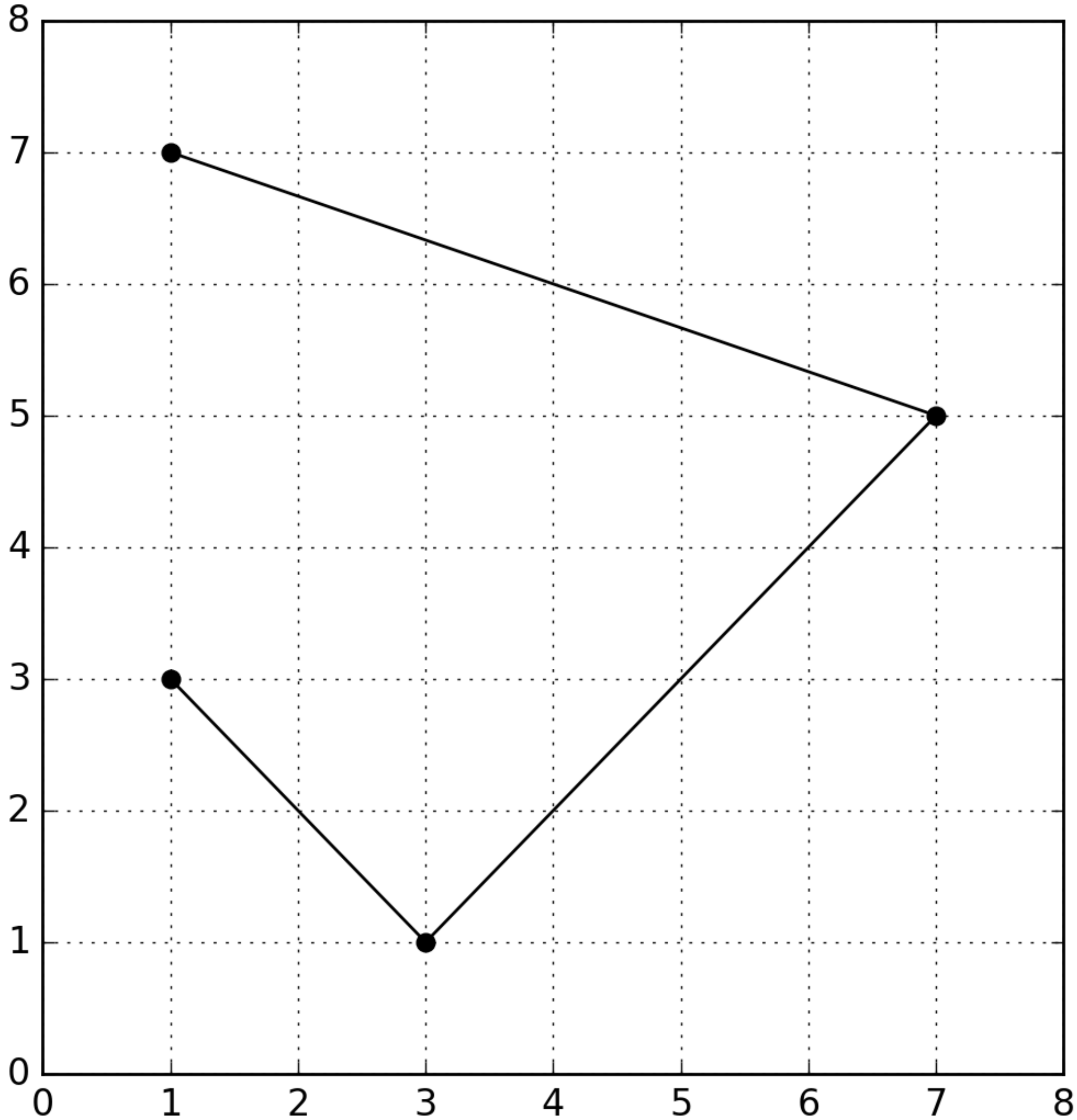
Ausgabe: [_ , _ , _ , _]

```
a = [1, 2, 3, 4, 5, 6]
a = list(filter(lambda x: x % 3 == 0, a))
print(a)
```

Ausgabe: [_ , _]

Aufgabe 5 — Bézierkurven (9 Punkte)

a) Bestimmen Sie graphisch zum gegebenen Kontrollpolygon den Punkt der zugehörigen Bézierkurve mit Parameter $t = \frac{1}{2}$.



b) Lesen Sie aus Aufgabe a) die Koordinaten von vier Punkten ab, die als Kontrollpolygon genau die untere Hälfte der ursprünglichen Bézierkurve beschreiben.

$(x_1, y_1) = (\quad , \quad)$ $(x_2, y_2) = (\quad , \quad)$ $(x_3, y_3) = (\quad , \quad)$ $(x_4, y_4) = (\quad , \quad)$

c) Nennen Sie drei Formeigenschaften von Bézierkurven.

-
-
-

Aufgabe 6 — Iterative Lösungsverfahren (10 Punkte)

Gegeben sei folgendes Gleichungssystem $Ax = b$:

$$\begin{bmatrix} 2 & 0 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 10 \\ 2 \\ 5 \end{bmatrix}$$

a) Führen Sie einen Iterationsschritt des Gauß-Seidel-Verfahrens mit dem Startwert $[1, 2, 3]^T$ aus. Der Rechenweg muss erkennbar sein.

b) In dieser Aufgabe sind Matrizen im CRS-Format gespeichert. Die Indizierung beginnt bei 0, sodass die Matrix:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 2 & 0 & 3 & 4 \\ 0 & 5 & 0 & 6 \\ 7 & 8 & 0 & 9 \end{bmatrix}$$

folgende CRS-Darstellung hat:

$$\begin{aligned} \texttt{val} &= [1, 2, 3, 4, 5, 6, 7, 8, 9] \\ \texttt{col\_ind} &= [0, 0, 2, 3, 1, 3, 0, 1, 3] \\ \texttt{row\_ptr} &= [0, 1, 4, 6, 9] \end{aligned}$$

Schreiben Sie eine Funktion die für gegebene Indizes i, j den richtigen Wert der Matrix zurückgibt, wenn die Darstellung lediglich im CRS-Format gegeben ist. Sie dürfen annehmen, dass die Indizes gültig sind.

```
import numpy as np

def indices(val, col_ind, row_ptr, i, j):
```

Aufgabe 7 — Lineare Algebra (14 Punkte)

Von der Matrix A sei die Singulärwertzerlegung bekannt:

$$A = \frac{1}{5}\begin{bmatrix} 15 & -8 \\ -20 & -6 \end{bmatrix} = \begin{bmatrix} \frac{3}{5} & \frac{4}{5} \\ -\frac{4}{5} & \frac{3}{5} \end{bmatrix} \begin{bmatrix} 5 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

a) Bestimmen Sie den Rang, die Konditionszahl κ_2 bezüglich der Spektralnorm $||.||_2$ und die Eigenwerte von AA^T.

b) Geben Sie die Definitionen der Spektralnorm $||\cdot||_2$, der Zeilensummennorm $||\cdot||_\infty$ und der Frobeniusnorm $||\cdot||_F$ an und bestimmen Sie $||A||_2$, $||A||_\infty$ und $||A||_F$.

c) Gegeben sei nun die Matrix $B = \frac{1}{5}\begin{bmatrix} 12 & -12 \\ -16 & -9 \end{bmatrix}$ Sowie die matrizen $U = \frac{1}{5}\begin{bmatrix} 3 & 4 \\ -4 & 3 \end{bmatrix}$ und $V^T = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$ der Singulärwertzerlegung. Bestimmen Sie eine die fehlende Matrix Σ der Singulärwertzerlegung von B.

Aufgabe 8 — Nichtlineare Optimierung (10 Punkte)

Sie versuchen, das Minimum der folgenden Funktion zu bestimmen:

$$F(x, y) = -x^3 + x^2 y - 2xy^2 + 4x + 3$$

a) Berechnen Sie die Jacobi- und die Hessematrix von $F(x, y)$.

b) Führen Sie einen Schritt des Gradientenabstiegs-Verfahren durch. Verwenden Sie dazu die Schrittweite $t_0 = 1$ und den Startwert $[x_0, y_0] = [2, 1]$.

c) Führen Sie für das gleiche Problem einen Schritt des Newton-Verfahrens durch. Wählen Sie als Startwert erneut $[x_0, y_0] = [2, 1]$. **Hinweis:** Gesucht ist das Minimum von F, nicht die Nullstellen.

Aufgabe 9 — Random way of life (7 Punkte)

Implementieren Sie den Zeitschritt *next_step* eines 1D zellulären Automaten der für ein gegebenes Gitter das zugehörige neue Gitter des nächsten Zuges mit folgenden Regeln berechnet:

- Eine lebendige Zelle, deren beide Nachbarn leben, stirbt.
- Mit einer 10% Wahrscheinlichkeit erwacht eine tote Zelle zum Leben.

Die Randbedingungen seien dabei periodisch, d.h., eine Zelle am linken Rand hat als linken Nachbarn die zugehörige Zelle am rechten Rand. Der andere Randfall ist analog.

Das Gitter *grid* ist ein *numpy* array. Eine tote Zelle hat den Wert 0, eine lebendige Zelle den Wert 1.

Hinweis: Die Funktion `random()` gibt eine zufällige Gleitkommazahl zwischen 0.0 und 1.0 zurück.
Die gepunkten Linien sind als Schreibhilfe gedacht, um die Python Indentierung korrekt hinzuschreiben.

```
import numpy as np
from random import random

def next_step(grid):
```