Lehrstuhl für Wissenschaftliches Rechnen
Fakultät für Informatik
Technische Universität München

Persönlicher Sticker

S5479

Bestätigung der Verhaltensregeln
Hiermit versichere ich, dass ich diese Klausur ausschließlich unter Verwendung der unten aufgeführten Hilfsmittel selbst löse und unter meinem Namen abgebe.

Unterschrift oder vollständiger Name, falls keine Stifteingabe verfügbar

Numerisches Programmieren

Klausur:	IN0019 / Endterm	**Datum:**	Dienstag, 2. März 2021
Prüfer:	Prof. Dr. Hans-Joachim Bungartz	**Uhrzeit:**	14:15 – 15:45

Bearbeitungshinweise

- Diese Klausur umfasst **14 Seiten** mit insgesamt **5 Aufgaben**.
  Bitte kontrollieren Sie jetzt, dass Sie eine vollständige Angabe erhalten haben.
- Die Gesamtpunktzahl in dieser Prüfung beträgt 58 Punkte.
- Verwenden Sie für Ihre Lösungen nur die dafür vorgesehenen Lösungsfelder!
- Als Hilfsmittel sind zugelassen:
  - beliebige Lehrmaterialien, aber keine Tools, die die Lösunswege und Lösungen direkt berechnen. Alle Lösungen und Lösungswege müssen selbstständig erarbeitet werden!
- Mit * gekennzeichnete Teilaufgaben sind ohne Kenntnis der Ergebnisse vorheriger Teilaufgaben lösbar.
- **Es werden nur solche Ergebnisse gewertet, bei denen der Lösungsweg erkennbar ist.** Auch Textaufgaben sind **grundsätzlich zu begründen**, sofern es in der jeweiligen Teilaufgabe nicht ausdrücklich anders vermerkt ist.
- Am Ende der Klausur finden Sie zusätzlichen Platz für Rechungen und Lösungen. Diese Seite wird nur dann korrigiert, wenn ein expliziter Vermerk bei der entsprechenden Aufgabe steht!

Hörsaal verlassen von __________ bis __________ / Vorzeitige Abgabe um __________

☐ Klausur leer

Aufgabe 1 Gleitkommazahlen, Kondition und Stabilität (13 Punkte)

0
1
2
3

a) **Binäres IEEE Float Format**

In dieser Aufgabe wird die Zahl $z = -\frac{32}{5}$ ($= -6.4_{10}$) in das binäre IEEE Float Format umgewandelt.

Berechnen Sie die Zahl als Binärzahl. Markieren Sie eine mögliche Periodizität in der Berechnung mittels Klammern (z.B.: $11.\overline{010}_2 = 11.(010)_2$).

32/5 = 6 + 2/5 -> 110
4/5 = 0
8/5 = 1 + 3/5
6/5 = 1 + 1/5
2/5 = 0 -> Perodizität
-32/5 = -110.(0110)

0
1
2
3

b) Wandeln Sie nun die Zahl in das binäre IEEE Float Format um. Geben Sie das Ergebnis als 32-Bit Bitfolge an und markieren Sie Vorzeichen, Exponent und Mantisse. Der Rechenweg muss erkenntlich sein!

Hinweis: Das IEEE Float Format verwendet 1 Bit für das Vorzeichen, 8 Bit für den Exponenten und 23 Bit für die Mantisse. Runden Sie wie in der Übung besprochen. Schreiben sie bei den Bitfeldern die ganze Zahl aus und kürzen Sie diese nicht ab!

Normalisierung: -110.(0110) = 2^2 * -1.10(0110)
Zahl ist negativ -> V = 1
Exponent ist 2 -> E = 2 + 127 = 129 -> 1000,0001
Manitsse, 23 Nachkommastellen ohne führende 1 -> M = 1001,1001,1001,1001,1001,101 (letzte Bit wegen Rundung)

Die Zahl ist also:
1|10000001|10011001100110011001101
V E M

Gegeben ist die Funktion $f(x) = 1 - e^x$.

c)* Berechnen Sie die Konditionszahl der Funktionsauswertung ($f(x)$) in Abhängigkeit von x.

0 1

c(x) = |(x * f'(x)) / (f(x))| = |(x * (-e^x)) / (1 - e^x)| = |(x * e^x) / (1 - e^x)|

d)* Analysieren Sie mithilfe der Epsilontik die Stabilität der Funktion. Die Auswertung von e^x erzeugt nur einen relativen Fehler $\varepsilon < \varepsilon_{\text{Ma}}$ (Maschinengenauigkeit)

0 1 2 3 4

sp(x) = (1 - e^x*(1 + E1))(1 + E2) = 1 - e^x - e^x*E1 + E2 - e^x*E2 = f(x) + f(x)*E2 - e^x*E1
s(x) = |(sp(x) - f(x)) / f(x)| = |E2 - (e^x*E1)/(1 - e^x)|

e) Was können Sie zur Kondition und Stabilität der Funktion für $x \approx 0$ sagen?

0 1 2

Kondition: lim x->0 c(x) = 1 (zB nach l'Hospital), also gut
Stabilität lim x->0 s(x) = oo, also schlecht

Aufgabe 2 Interpolation (9 Punkte)

Beim Newton-Verfahren berechnen wir die Einträge in einem Dreiecksschema:

$$
\begin{array}{c|c|ccccccc}
x_i & i \backslash k & 0 & & 1 & & 2 & & \cdots \\
\hline
x_0 & 0 & c_{0,0} = y_0 & \rightarrow & c_{0,1} & \rightarrow & c_{0,2} & \rightarrow & \cdots \\
 & & & \nearrow & & \nearrow & & & \\
x_1 & 1 & c_{1,0} = y_1 & \rightarrow & c_{1,1} & \rightarrow & \vdots & & \\
 & & & \nearrow & & & & & \\
x_2 & 2 & c_{2,0} = y_2 & \rightarrow & \vdots & & & & \\
\vdots & \vdots & \vdots & & & & & &
\end{array}
$$

a) In dieser Aufgabe soll zu den Punkten

$$P_0 = (-2, 1),\ P_1 = (0, -2) \text{ und } P_2 = (2, 3)$$

ein Interpolationspolynom mit dem Newton-Verfahren bestimmt werden. Berechnen Sie alle $c_{i,k}$ Werte. Geben Sie Ihr Ergebnis in der Form '$c_{i,k}$ = <Berechnung>' bzw. 'c_i,k = <Berechnung>' für alle relevanten i und k Bereiche an. Geben Sie auch den Rechenweg an (Formel mit eingesetzten Werten)!

c0,0 = y0 = 1
c1,0 = y1 = -2
c2,0 = y2 = 3

c0,1 = (c1,0 - c0,0) / (x1 - x0) = (-2 - 1) / (0 - (-2)) = -3/2 = -1.5
c1,1 = (c2,0 - c1,0) / (x2 - x1) = (3 - (-2)) / (2 - 0) = 5/2 = 2.5

c0,2 = (c1,1 - c0,1) / (x2 - x0) = (2.5 - (-1.5)) / (2 - (-2)) = 4/4 = 1

b) Geben Sie das Polynom $p(x)$ an, das durch das oben berechnete Newton-verfahren bestimmt wurde. Das Ausmultiplizieren der Komponenten ist nicht nötig. Werten Sie **zusätzlich** das Polynom an der Stelle 1 aus ($p(1)$).

p(x) = c0,0 + c0,1(x - x0) + c0,2*(x- x0)(x - x1) = 1 - 2(x + 2) + (x + 2)x

p(1) = 1 - 2(1 + 2) + (1 + 2)*1 = 1 - 6 + 3 = -2 c)* Was ist die Komplexität der Durchführen des Newton-Verfahrens mit N Punkten und was ist die Komplexität, um das entstehende Polynom auszuwerten? Gehen Sie von einer optimalen Implementierung aus.

0 1 2

Durchführen: O(n^2)
Auswerten: O(n)

d)* Sie wollen die Ausbreitung der Coronapandemie anhand der täglichen Fallzahlen graphisch veranschaulichen. Hierfür entnehmen Sie der Presse einen Datensatz mit den täglich gemeldeten Fallzahlen (ohne Nachmeldungen) in München an den letzten 300 Werktagen während der Pandemie. Da am Wochenende keine Zahlen übermittelt werden, wollen Sie die fehlenden Werte mittels Interpolation ermitteln. Eignet sich für dieses Szenario das Newton-Verfahren zur Interpolation? Begründen Sie ihre Antwort und nennen Sie ein geeignetes alternatives Verfahren, falls das Newton-Verfahren nicht geeignet sein sollte.

0 1 2

Nein, denn die Rechungsfehler bei dem Newton-Verfahren vergrößert sich bei mehreren Stützstellen, bei einem Datensatz mit 300 Stellen wird die Fehler enorm. Da könnte man kubische Splines empfehlen, wo die Fehler unabhängig von der Anzahl von Stellen ist.

Aufgabe 3 Quadratur (10 Punkte)

In dieser Aufgabe wollen wir eine Quadraturformel herleiten, bei der nur auf einer der beiden Integralgrenzen ein Stützpunkt liegt. Für eine unbekannte Funktion $g(x)$ wollen Sie das Integral

$$\int_0^1 g(x)dx \tag{1}$$

berechnen. Hierfür erhalten Sie durch Messungen die Funktionswerte an den Stellen $x_0 = 0$, $x_1 = 1/4$, $x_2 = 1/2$. Sie wollen hiermit eine Quadraturformel aufstellen, indem Sie das Interpolierende Polynom durch die Funkionswerte integrieren.

0 1

a) Welchen Grad hat das Polynom, dass die 3 gegebenen Punkte interpoliert?

Grad 2

0 1 2 3 4 5

b)* Wir wollen nun die Quadraturgewichte bestimmen. Arbeiten sie hierfür mit allgemeinen Funktionswerten y_0, y_1 und y_2 und bestimmen Sie die Gewichte w_0, w_1 und w_2. Verwenden Sie entweder die Methode der Unbestimmten Koeffizienten (**Tipp:** Hier sind nur die Gewichte gesucht, die Stützstellen sind bereits gegeben) oder stellen Sie explizit das Interpolationspolynom auf und integrieren es. Die Integralgrenzen sind $a = 0$ und $b = 1$.

g(x) = w0 + w1*x + w2*x^2

g(0) = w0 = y0
g(1/4) = w2/16 + w1/4 + w0 = y1
g(1/2) = w2/4 + w1/2 + w0 = y2

LGS erstellen:
(0   0   1 | y0)   (0   0   1 | y0   )
(1/16 1/4 1 | y1) => (0   -1/2 -3 | y2 - 4y1)
(1/4   1/2 1 | y2)   (1/4 1/2 1 | y2   )

w0 = y0
w1 = -2((y2 - 4y1) + 3w0) = -2y2 + 8y1 - 6y0
w2 = 4(y2 - w0 - w1/2) = 4(y2 - y0 - (-y2 + 4y1 - 3y0)) = 8y2 - 16y1 + 16y0

Integrieren:

S(0,1) w0 + w1*x + w2*x^2 = [w0*x + w1*x^2/2 + w2*x^3/3](1, 0) =w0 + w1/2 + w2/3 = y0 + (-y2 + 4y1 - 3y0) + (8y2 - 16y1 + 16y0)/3

c) Welches Problem besteht bei der berechneten Quadraturformel in Bezug auf die Kondition des Quadraturproblems?

0 1

Negative Gewichte sind möglich, das kann die Kondition schlechter machen

d)* Welches Verfahren aus der Vorlesung und Übung erreicht den maximalen Polynomgrad für ein allgemeines Quadraturproblem mit freier Stützstellenwahl? Nennen Sie das Verfahren und den Polynomgrad, bis zu dem es bei N Stützstellen exakt ist! Erklären sie zusätzlich die methodische Idee durch die dieser hohe Grad ermöglicht wird!

0 1 2 3

Gaußquadratur erreicht bei N Stützstellen schon ein Grad von 2N - 1
Dies ist möglich, weil man nicht nur die Gewichter auswählt, sondern auch die Stützstellen, was das Grad quasi "verdoppelt"

Seite leer

Aufgabe 4 Eigenwerte (13 Punkte)

Gegeben ist die symmetrische Matrix A mit Eigenwerten $\lambda_1 = 10$, $\lambda_2 = 3$ und $\lambda_3 = 2$.

0 1 2

a) Können alle Eigenvektoren zu diesen Eigenwerten mit der normalen Poweriteration (inklusive Shift) ermittelt werden? Begründen Sie ihre Antwort.

Nein, die normale Poweriteration konvergiert zum (nach dem Shift) betragsmäßig größten Eigenwert, und es ist unmöglich, den mittleren Eigenwert den größten zu machen, auch mit Shifts. Die anderen sind jedoch erreichbar.

0 1 2 3

b)* Geben Sie zu jedem Eigenwert einen möglichen Shift μ an, für den die Iteration gegen den zugehörigen Eigenvektor konvergiert. Falls es für einen Eigenwert keinen passenden Shift gibt, dann nennen Sie ein alternatives Verfahren, mit dem der Eigenvektor ermittelt werden kann. Es muss keine Begründung angegeben werden. Geben Sie für alle Ihre Lösungen auch die Konvergenzgeschwindigkeit K an.

L1: kein Shift nötig, also 0, K = |L2 / L1| = 0.3
L2: unmöglich, aber mit zB Inverse Power Iteration erreichbar
L3: mit Shift = 7, K = |(L2 - 7) / (L3 - 7)| = 0.8

0 1 2

c)* Bei welchem Shift μ gibt es die größte Konvergenzgeschwindigkeit für den Eigenvektor der zum Eigenwert $\lambda_1 = 10$ gehört? Geben Sie μ und die daraus resultierende Konvergenzgeschwindigkeit K an. Es ist keine Begründung nötig.

Bei einem Shift von 2
K = |(L2 - 2) / (L1 - 2)| = 1/8

0 1

d)* Berechnen Sie die Kondition der Matrix A. Geben Sie den Rechenweg an!

cond(A) = |(größter Eigenwert) / (kleinster Eigenwert)| = 10/2 = 5 e)* Wie kann die (inverse) Poweriteration allgemein genutzt werden um die Matrixkondition einer symmetrischen Matrix anzunähern? Beschreiben Sie das Vorgehen.

0 1 2

Da die Kondition eines Matrixes sich durch den Verhältnis seiner größten bzw. kleinsten Eigenwerten berechnen lässt, kann man Poweriterationen benutzen, um diese Eigenwerte zu finden: mit dem normalen findet man den größten, mit dem inversen den kleinsten, und divideren.

f)* Gegeben ist die Matrix $B = \begin{pmatrix} 4 & 2 & 1 \\ 2 & 7 & 0 \\ 1 & 0 & 2 \end{pmatrix}$. Welche der folgenden Eigenschaften treffen auf die Matrix B zu und welche nicht? Begründen Sie Ihre Antworten kurz!

0 1 2 3

1. Diagonaldominant
2. Symmetrisch positiv definit (spd)
3. Orthogonal

Diagonaldominant: ja, die Diagonaleinträge sind größer als die Summen der nicht D.einträge ihrer Spalten
spd: ja, denn A = A^T und alle Einträge positiv oder = 0
Orthogonal: nein, die erste Zeile * die zweite Zeile ist ungleich null

Aufgabe 5 Programmieraufgabe: Abstiegsverfahren für Lineare Gleichungssysteme (13 Punkte)

In dieser Aufgabe wird ein Lineares Gleichungssystem $Ax = b$ durch das Verfahren des steilsten Abstiegs gelöst. Jede Iteration des Verfahrens des steilsten Abstiegs entspricht dabei folgenden drei Schritten:

1. Berechnung des aktuellen Residuums r
2. Berechnung der optimalen Schrittweite α
3. Berechnung des aktuellen Zwischenergebnisses x

0 1 2

a) Implementieren Sie die Hilfsfunktionen `computeResidual()` für Schritt 1

```
/**
 * Compute the residual for the current guess
 *
 * @param A    system matrix
 * @param b    right hand side vector
 * @param x    current solution guess
 * @return r   computed residual
 */
private double[] computeResidual(double[][] A, double[] b, double[] x) {
    int length = x.length;
```

```
double[] r = new int[length];
for (int i = 0; i < length; i++) {
  r[i] = b[i];
  for (int j = 0; j < length; j++)
    r[i] -= A[i][j] * x[j]
}
return r;
```

```
}
```

b)* und `computeStepSize()` für Schritt 2

```
/**
 * Compute the required stepsize for the given direction
 *
 * @param A        system matrix
 * @param r        residual
 * @return alpha   computed step size
 */
private double computeStepSize(double[][] A, double[] r) {
    int length = r.length;
```

```
double alpha = 0;
double div = 0;
for (int i = 0; i < length; i++) {
  alpha += r[i] * r[i];
  double sum = 0;
  for (int j = 0; j < length; j++)
    sum += A[i][j] * r[j];
  div += r[i] * sum;
}
return alpha / div;
```

```
}
```

0 1 2 3 4

c)* Ergänzen Sie nun den unten stehenden Quelltext mithilfe der Hilfsfunktionen. Achten Sie dabei auf eine geeignete Abbruchbedigung der Iteration. Eine Rekursion soll nicht benutzt werden.

```java
public class SteepestDescentSolver {

    /**
     * Solves Ax = b iteratively via steepest descent
     *
     * @param A   system matrix
     * @param b   right hand side vector
     * @param x0  initial value for iterative solution
     * @return x  solution of the system
     */
    public double[] solve(double[][] A, double[] b, double[] x0) {
        int length = x0.length;
        double[] x = x0.clone();
        double[] r = new double[length];
```

```
while(true) {
  r = computeResidual(A, b, x);
  double alpha = computeStepSize(A, r);
  for (int i = 0; i < length; i++)
    x[i] += alpha * r[i];
  bool finished = true;
  for (int i = 0; i < length; i++) {
    double sum = 0;
    for (int j = 0; j < length; j++)
      sum += A[i][j] * x[j];
    if (sum != b[i])
      finished = false;
  }
  if (finished)
    break;
}
```

```java
        return x;
    }
}
```

d)* Das Verfahren des Steilsten Abstiegs hat gewisse Nachteile bezüglich der Konvergenzgeschwindigkeit. Nennen Sie einen dieser Nachteile und erläutern Sie, wie man diesen Nachteil umgehen kann?

Die Konvergenzrate ist schlecht, auch wenn man relativ nah am Ergebnis kommt. Diese Problem kann man lösen, in dem man auch nach besseren Suchrichtungen sucht..

Zusätzlicher Platz für Lösungen. Markieren Sie deutlich die Zuordnung zur jeweiligen Teilaufgabe. Vergessen Sie nicht, ungültige Lösungen zu streichen.

☐ Seite leer