Online-Prüfung im Fach
Computational Engineering

Für M-TI (FB 6)
WS 2021/22, 1. Prüfungszeitraum
Prüfungstermin am 02.02.2022
Prüfer: Prof. Dr. Marc Kirch

Berliner Hochschule für Technik

Bearbeitungszeit: 120 Min. (Prüfungszeit) + 10 Min. (Extrazeit) = 130 Minuten

Bewertungsrahmen: Für eine 100%-Wertung benötigen Sie nur 54 der maximal 60 in dieser Klausur zu erreichenden Punkte. Eine Wertung von über 100% ist dabei nicht möglich.
Bestanden haben Sie ab einer Punktzahl von 27.

Eidesstattliche Erklärung: Mit meiner Unterschrift bestätige ich mit der Prüfungsform einverstanden zu sein, die rechtzeitig zuvor kenntlich gemachten Regeln zur Durchführung der Prüfung erhalten und verstanden zu haben und die Prüfung eigenständig und ohne Mithilfe von weiteren Personen bearbeitet zu haben:

Name: _____ Matrikelnr.: _____

Unterschrift: _____

Hinweis: Wenn Sie keinen Drucker zur Verfügung haben, um das Deckbklatt auszudrucken, dann können Sie diesen Teil auch handschriftlich auf ein Extrablatt kopieren und ausfüllen.

Aufgabe:	1	2	3	4	$\sum$
Punkte:					

Note:	
Berlin, den	

(Unterschrift Prof. Kirch)

Weitere Hinweise zur Bearbeitung

- Diese Prüfung ist eine Prüfungsleistung im Sinne der RSPO. Mit dem endgültigen Hochladen Ihrer Lösungen im Prüfungsportal, zählt dies im Rahmen der zum Zeitpunkt der Prüfung geltenden Regelung als zu bewertender Prüfungsversuch.
- Die auf dem Deckblatt angegebene Bearbeitungszeit ist die Zeitspanne zwischen der Verfügbarmachung der Aufgaben und der Schließung der Möglichkeit die Lösungen hochzuladen. Das Portal zum hochladen der Aufgaben schließt automatisch und nach der Schließung sind keine weiteren Abgaben möglich und werden auch auf anderem Wege nicht akzeptiert. Probleme mit der Hochladeprozess müssen innerhalb der Bearbeitungszeit kommuniziert werden.
- Außer der direkten oder indirekten Hilfe durch Zusammenarbeit mit anderen Personen, dürfen Sie alle möglichen Hilfsmittel verwenden.
- Sie können ihre Lösungen als m-Files oder als reine Textfiles hochladen.
- Die Programme müssen für eine volle Punktzahl fehlerfrei laufen und alle in dwer Aufgabenstellung geforderten Aufgaben abarbeiten.
- Im Sinne der Aufgabenstellung überflüssige Code-Bestandteile führen ggfs. zu Punktabzug. Kommentare sind hiervon ausgenommen.
- Schreiben Sie ihre Programme so einfach und linear wie möglich und nur so komplex wie nötig.

Aufgabe 1 (15 Punkte)

Geben Sie ein Matlab **Skript** an, in welchem Sie eine **Laufzeitvergleich** der Matlabbefehle `inv` und des `Backslashoperators` durchführen. Gehen Sie dabei wie folgt vor:

1. Sie füllen eine quadratische Matrix $\mathbf{A}_N$ der Dimension $N = 2^k$ für $k = 2, 3, \ldots, 10$ mit Zufallszahlen, die im Intervall $[-1, 1]$ gleichverteilt sind.
2. Für jedes N berechnen Sie mindestens $R = 20$ mal (jeweils für eine neue zufällige Matrix) die Inverse Matrix von $\mathbf{A}_N$ auf zwei verschiedene Arten:
   (a) `inv`$(\mathbf{A}_N)$
   (b) $\mathbf{A}_N \backslash \mathbf{E}$ (worin $\mathbf{E}$ die N-dimensionale Einheitsmatrix ist.)
3. Zeichnen Sie für beide Rechnungen aus Schritt 2 jeweils die Rechenzeiten auf. Bilden Sie für jedes N einen Mittelwert der Rechenzeiten über die R Wiederholungen. Sie haben dann einen Vektor $T_a(N)$ und einen Vektor $T_b(N)$ der jeweiligen mittleren Rechenzeiten.
4. Fertigen Sie eine Graphik an, in der $T_a(N)$ und $T_b(N)$ dargestellt wird.
5. **Approximieren** Sie den Verlauf der mittleren Rechenzeiten für a und b **jeweils** durch ein Polynom vom Grad 2. Sie dürfen hierzu eine implementierte Matlab-Funktion verwenden. Die Graphen der Polynome werden ebenfalls in der Graphik dargestellt. Die Graphik ist aussagekräftig formatiert.

Aufgabe 2 (15 Punkte)

Gegeben ist das lineare Gleichungssystem (LGS)

$$\begin{aligned} -5x_1 + 2x_2 - 4x_3 &= 0 \\ 3x_1 + 6x - 5x_3 &= -8 \\ -x_1 + 8x_2 - 9x_3 &= -10 \end{aligned}$$

sowie folgende **Vorschrift für ein iteratives Lösungsverfahren**:

$$\vec{x}_{n+1} = \mathbf{D}^{-1} \cdot \left((\mathbf{D} - \omega \mathbf{A}) \cdot \vec{x}_n + \omega \vec{b} \right)$$

Darin ist $\mathbf{D}$ die Matrix, die nur die Diagnonale der Koeffizientenmatrix $\mathbf{A}$ des LGS enthält und ω ist ein reeller Parameter.

Vorgegeben sei der **Startpunkt** der Iteration $\vec{x} = (10, 10, 10)^t$ und die exakte Lösung des LGS ist ebenfalls bekannt $\vec{x}_{exakt} = (-2, 3, 4)^t$.

Geben Sie ein Matlab Skript an, in welchem Sie das LGS näherungsweise mittels der angegebenen Iterationsvorschrift, für die Werte $\omega = 0.19, 0.20, 0.21, \ldots, 1.29$ lösen. Ermitteln Sie darin, wie viele Iterationsschritte $N(\omega)$ für jeden Wert von ω jeweils notwendig sind, bis für die iterative Lösung $\vec{x}_{N(\omega)}$ gilt

$$|\vec{x}_{N(\omega)} - \vec{x}_{exakt}| < 10^{-3}$$

Fertigen Sie einen Plot an, in dem Sie $N(\omega)$ über ω auftragen. Im Titel des Plots geben Sie den Wert von ω an, für den am wenigsten Iterationsschritte notwendig sind. Hinweis: für $\omega = 1$ ist die Iterationsvorschrift identisch mit der Jacobi-Iteration.

Aufgabe 3 (15 Punkte)

Gegeben ist das Anfangswertproblem (AWP)

$$y^3 - x^2 + xy^2y' = 0, \qquad y(1) = 2$$

für die unbekannte Funktion $y = y(x)$.

Geben Sie ein Matlab **Skript** an, in welchem Sie das AWP auf dem Intervall $x \in [1, 3]$ näherungsweise durch das im folgenden beschriebene **adaptive Einschrittverfahren** lösen:

1. Setze $k = 1$, $x_k = 1$, $x_e = 3$, $y_k = 2$, $TOL = 0.001$
2. Wenn $x_k \geq x_e$, dann STOP und gehe zu Schritt 7, sonst gehe zu Schritt 3.
3. Setze $h_k = 0.1$
4. Berechne den nächsten y-Wert **sowohl** mittels des **Euler-Verfahrens** y_{k+1}^{Euler}, als auch mittels der Regel von **Heun** y_{k+1}^{Heun}.
5. Berechne die Differenz $\delta_k = |y_{k+1}^{\text{Euler}} - y_{k+1}^{\text{Heun}}|$
6. Wenn
   (a) $\delta_k < TOL$, dann **akzeptiere die Lösung** $y_{k+1} = y_{k+1}^{\text{Euler}}$ und setze $x_{k+1} = x_k + h_k$, $k \to k + 1$ und gehe zu Schritt 2,
   (b) andernfalls setze $h_k \to 0.5h_k$ und gehe zu Schritt 4.
7. Plotte das Ergebnis y gegen x.

Der Plot im letzten Schritt ist aussagekräftig formatiert und die einzelnen errechneten Datenpunkte sind klar erkennbar.

Aufgabe 4 (15 Punkte)

Betrachten Sie einen **eindimensionalen Random walk auf einer Kreislinie** mit dem Radius eins. Die Kreisline ist dabei in $L = 100$ Teilstücke der Länge $\Delta x = \frac{2\pi}{L}$ unterteilt. Der Random Walker startet immer an einem fest vorgegebenen Teilstück auf der Kreislinie (12 Uhr-Position) und führt jeweils mit gleicher Wahrscheinlichkeit einen Schritt in ein benachbartes Teilstück, entweder im oder gegen den Uhrzeigersinn aus.

Fertigen Sie ein Matlab Skript an, in welchem Sie die **mittlere Entfernung des Random Walkers von seinem Ausgangspunkt entlang der Kreislinie** nach $n = 1, \ldots, N$ Schritten und die zugehörige **Standartabweichung** (matlab Befehle `mean` und `std`)) berechnen. Die Mittelung erfolgt dabei an jedem n über R Wiederholungen. Ihr Skript gibt einen Plot der mittleren Entfernung über n mit Fehlerbalken (maltlab Befehl `errorbar`) aus.

Zum testen nehmem Sie gern kleinere Werte, für die Simulation nimmt man $N > 2500$ und $R > 100$. Der Plot ist formatiert.
