Numerik 1, Studiengang Informatik, ZHAW

Übungsserie 4

Abgabe: gemäss Angabe Dozent

Fassen Sie Ihre Lösungen in einer ZIP-Datei *Name_ Vorname_ Gruppe_ S4.zip* zusammen. Laden Sie dieses File vor der nächsten Übungsstunde nächste Woche auf OLAT hoch. Funktionen müssen ausführbar sein und in den Kommentarzeilen soll ein Beispiel eines funktionierenden Aufrufs angegeben werden.

Aufgabe 1 (45 Minuten):

Betrachten Sie die folgenden Aussagen:

- Der Graph einer Exponentialfunktion $f(x) = c \cdot a^x$ in einem Koordinatensystem mit logarithmischer y-Achse ist eine Gerade.
- Der Graph einer Potenzfunktion $f(x) = c \cdot x^a$ ist eine Gerade, wenn man beide Koordinatenachsen logarithmisch wählt.

a) Beweisen Sie diese beiden Aussagen zuerst für sich selbst analytisch (Tipp: berechnen Sie $y = \log f(x) = ...$). Geben Sie dann den y-Achsenabschnitt und Steigung der untenstehenden Funktionen in dem Koordinatensystem an, in dem der Funktionsgraph eine Gerade ist. Scannen Sie Ihre Lösung in *Name _ Vorname_ Gruppe_ S4_ Aufg1a.pdf.*

- $f(x) = \frac{5}{\sqrt[3]{2x^2}}$
- $g(x) = 10^5 \cdot (2\mathrm{e})^{-x/100}$
- $h(x) = (\frac{10^{2x}}{2^{5x}})^2$

b) Um das Erstellen solcher Graphiken zu unterstützen, stellt MATLAB die Anweisungen `logspace`, `semilogx`, `semilogy` und `loglog` zur Verfügung. Schreiben Sie ein Skript *Name_ Vorname_ Klasse_ S4_ Aufg2b*.m, welches Ihnen die Graphen der obigen Funktionen als Geraden darstellt, jeweils für $0 \leq x \leq 100$.

Aufgabe 2 (ca. 45 Min.):

Implementieren Sie das Bisektionsverfahren in einer MATLAB-Funktion
`[root,xit,n] = Name_Vorname_Gruppe_S4_Aufg2(func,a,b,tol)`, welches Ihnen eine Nullstellen einer beliebigen Funktion `func` auf dem Intervall [a,b] mit einer Genauigkeit von mindestens `tol` berechnen soll bzw. eine Fehlermeldung ausgibt, falls keine gefunden werden kann. Die Näherung für die Nullstelle soll in die Variable `root` geschrieben werden, Iterationswerte x_k sollen in den Vektor `xit` geschrieben werden, die Anzahl benötigter Iterationen in die Variable `n`. Als Abbruchkriterium können Sie $\mid a_k - b_k \mid \leq tol$ verwenden.

Bemerkung:

Die Funktion `func` können sie als anomye Funktion interaktiv defnieren, also z.B.

```
>> func = @(x) cos(x).*sin(x)
```

```
>> [root,xint,n] = Name_Vorname_Gruppe_S4_Aufg2(func,a,b,tol)
```

oder auch direkt in den Funktionsaufruf integrieren:

```
>> [root,xint,n] = Name_Vorname_Gruppe_S4_Aufg2(@(x) cos(x).*sin(x),a,b,tol)
```

Aufgabe 3 (ca. 45 Min.):

Schreiben Sie ein Skript `Name_Vorname_Gruppe_S4_Aufg3.m`, welches Ihnen unter Verwendung ihrer Funktion aus Aufgabe 2 für die folgende Gleichung und Startwerte

$$x^2 - 2 \quad = \quad 0, \quad a = 0, b = 2$$

- den absoluten Fehler ihrer Näherungswerte `xit` als Funktion der Anzahl durchgeführten Iterationen mit semilogy plottet für `tol` $\in \left\{10^{-15}, 10^{-16}\right\}$;
- die Anzahl benötigter Iterationen `n` als Funktion der geforderten Toleranz `tol` mit loglog plottet für `tol` $\in \left\{10^{-1}, 10^{-2}, ..., 10^{-20}\right\}$.

Interpretieren Sie die Ergebnisse unter Berücksichtigung der Erkenntnisse von Aufgabe 1 und schreiben Sie diese als Kommentare in Ihr Skript.