MATLAB-Vorbereitungsangebot

In den praktischen Aufgaben zur Vorlesung „Grundlagen der numerischen Mathematik und Optimierung“ wird MATLAB benutzt. Wenn Sie noch keine Erfahrungen mit MATLAB oder Octave haben, nutzen Sie die erste Woche, um sich damit vertraut zu machen.

Zu Hause Arbeiten

Um Ihre Aufgaben zu Hause bearbeiten zu können, können Sie Matlab auf Ihrem Rechner installieren. Alle Informationen dazu finden Sie unter

`https://www.cms.hu-berlin.de/de/aktuelles-und-veranstaltungen/matlab-und-simulink-kostenlos-fuer-hu-angehoerige-1`

Falls Sie umfangreichere Rechnungen durchführen wollen (für die Übunsgaufgaben im Allgemeinen nicht nötig), können Sie von zu Hause aus via ssh in der Uni arbeiten und MATLAB mittels `matlab -nodesktop` starten. Bitte verwenden Sie hierbei die speziell dafür bereitgestellten Server (alpha, beta, gamma, delta und epsilon).

Octave

Eine freiverfügbare Alternative zu MATLAB ist Octave. Die Syntax von Octave ist mit der von MATLAB nahezu identisch. Octave ist open source-Software und kann kostenfrei heruntergeladen und installiert werden. In vielen Linux-Distributionen kann Octave direkt mit Hilfe des Paketmanagers ausgewählt und installiert werden.

Erste Schritte

1. Starten Sie MATLAB. Im Pool des Institutes für Mathematik ist dies mit dem Befehl `matlab` möglich. Mittels Tabulatortaste können Sie sich die verschiedenen installierten Versionen anzeigen lassen.

2. Machen Sie sich mit der graphischen Oberfläche vertraut.

3. Suchen Sie sich im Internet ein Tutorial oder nutzen sie die verlinkten Einführungen auf der Numerikwebseite.

4. Rufen Sie die MATLAB-Hilfe auf und informieren Sie sich über Befehle wie `lu` (`help lu`) und \ (backslash, `help mldivide`). Nutzen Sie die interne Hilfe von MATLAB, um Sprachelemente zu finden und zu verstehen.

5. Erstellen Sie sich zur Übung mit dem integrierten Editor ein MATLAB-Skript (Folge von Befehlen) und arbeiten Sie diese ab.

Stellen Sie sich kleine Aufgaben, um den Umgang mit MATLAB zu üben. Die folgenden Aufgaben bieten Ihnen dazu ebenfalls Gelegenheit.

Householder Spiegelung

Lesen Sie einen Vektor v ein und berechnen Sie, falls v ungleich 0, eine Householder Spiegelung (siehe Householder Verfahren)

$$H = I - 2vv^T/(v^Tv) \qquad (I = \text{Einheitsmatrix})$$

Überprüfen Sie die Orthogonalitätseigenschaft von H numerisch.

Lineares Gleichungssystem

Erzeugen Sie mit `rand` eine zufällige Matrix A und einen zufälligen Spaltenvektor b (z.B. der Größe 3). Lösen Sie nun das Gleichungssystem $Ax = b$

1. mit dem Operator \, und
2. indem Sie die Matrix mit dem Befehl `lu` faktorisieren, und das Gleichungssystem anschließend mittels Vorwärts- und Rückwärtssubstitution lösen (`for`-Schleife!).

Schreiben Sie eine Funktion

```
x = LinSolve(A,b)
```

die ein Gleichungssystem mittels `lu` löst, wie oben beschrieben. Die Dimension des Systems können Sie mittels `length(b)` bestimmen.

Plotten

In MATLAB können Funktionen auf verschiedenste Weisen erklärt werden. Informieren Sie sich bitte mittels `help function_handle`, wie einfache Funktionen implizit erklärt werden können und schreiben Sie anschließend eine Funktion

```
UnitPlot(f,n)
```

welche eine gegebene Funktion f auf n gleichverteilten Stützstellen im Einheitsintervall auswertet (for-Schleife) und das Resultat anschließend mittels `plot` am Bildschirm darstellt. Achten Sie auf potentielle Fehlerquellen und fangen diese ggf. ab (z.B. mittels `error`).

Testen Sie Ihre Funktion an einigen selbstgewählten Beispielen.

Fibonacci-Zahlen

Funktionen können sich in MATLAB auch selbst aufrufen. Programmieren Sie eine Funktion

```
f = Fibonacci(n)
```

welche die n-te Fibonacci-Zahl f_n zurückgibt. Realisieren Sie dazu die rekursive Vorschrift

$$f_0 = 0, \quad f_1 = 1, \quad f_{n+2} = f_{n+1} + f_n$$

Prüfen Sie, ob f_{30} wirklich 832040 ist.