**Tag 1: Einführung in Programmierung und Benutzung von Matlab**

**A) Die Matlab-Oberfläche und Matlab als Taschenrechner**

**B) Vektoren und Matrizen**

**C) Grafische Darstellung von Vektoren**

**D) Hausaufgabe**

**A) Die Matlab-Oberfläche und Matlab als Taschenrechner:**

Matlab ist eine Programmierumgebung, die im wissenschaftlichen Bereich häufig genutzt wird. Im Rahmen dieses Kurses werden wir nur einen kleinen Teil der Möglichkeiten von Matlab ausschöpfen. Im ersten Schritt wollen wir uns erst einmal mit der Oberfläche von Matlab vertraut machen. (Die folgende Beschreibung bezieht sich auf Matlab 7, andere Versionen können geringfügig anders aussehen.)

Das wichtigste Fenster ist das große "Command window". Dort wird durch den Cursor >> angezeigt, dass Matlab bereit ist, Befehle entgegenzunehmen, die hier eingetippt werden können. Wenn Matlab gerade mit der Abarbeitung eines Programms beschäftigt ist, wird kein Cursor dargestellt und in der Leiste ganz unten steht "busy".

Unten links ist ein kleineres Fenster mit dem Namen "Command history". In diesem Fenster kann man seine letzten Eingaben in das Command window nachvollziehen. Wenn man eine Eingabe wiederholen möchte, drückt man die Pfeil-nach-oben-Taste. Dadurch lässt sich viel unnötige Tipparbeit sparen.

Darüber gibt es ein etwas größeres Fenster, das sich durch die beiden Knöpfe "Workspace" und "Current Directory" zwischen zwei Funktionalitäten umschalten lässt. Der "Workspace" ist die Gesamtheit aller zurzeit definierten Variablen. Diese werden als Liste in dem Fenster angezeigt. Eine Variable ist ein Name, der einen bestimmten Inhalt bezeichnet. Z.B. lässt sich nach der Definition a=1 mit a weiterarbeiten. Sein Inhalt lässt sich für Berechnungen benutzen (z.B. b=3\*a) oder ändern (z.B. a=a+1).

Schaltet man das Fenster um auf "Current Directory", zeigt es alle Dateien des aktuellen Verzeichnisses an. Dieses kann man mit dem kleinen pull-down-Menü ganz oben ändern. (Das Thema Speichern und Laden behandeln wir morgen.)

Außerdem findet man oben noch zwei Leisten. Sämtliche Funktionalitäten zu erklären würde über diesen Kurs hinausgehen, aber beide Leisten sind stückweise selbsterklärend. Eine sehr wichtige Funktion ist durch das Fragezeichen symbolisiert: Das Hilfefenster.

**Aufgaben:**

T1\_A1) Starten Sie Matlab und machen Sie sich mit den verschiedenen Fenstern vertraut. Starten Sie die Hilfe. Welche Möglichkeiten bietet sie?

T1\_A2) Benutzen Sie das Matlab Command Window als Taschenrechner. Probieren Sie die Grundrechenarten aus z.B. 7.32\*674.986 oder 100086+654.965-83.7 (jeweils mit Return abgeschlossen)

Wie unterscheidet sich die Antwort auf die Eingabe von 5\*10+2 von 5\*(10+2) und (5\*10)+2?

Achtung: in der amerikanischen Schreibweise wird ein Punkt verwendet, um die Stellen "hinter dem Komma" abzutrennen!

T1\_A3) Wenn Sie einige Rechnungen durchgeführt haben, tippen Sie ans ein - wie reagiert Matlab? Warum? Wie reagiert Matlab auf ans\*2? Was passiert, wenn Sie mehrfach hintereinander ans\*2 wiederholen? (Dazu können Sie die Pfeil-nach-oben-Taste verwenden.)

T1\_A4) Führen Sie mit a=4 eine Variable a ein. Beobachten Sie die Reaktionen in den verschiedenen Fenstern. Jetzt tippen Sie b=10; ein. Wie unterscheiden sich die Ausgaben? Wie reagiert Matlab auf die Eingabe a+b? Wie unterscheiden sich davon die Reaktionen auf c=a+b; ?

T1\_A5) Variablen können vom Benutzer beliebig eingeführt und mit Werten belegt werden. Es gibt aber auch Konstanten, die in Matlab vordefiniert sind, beispielsweise pi. Lassen Sie sich den Wert ausgeben.

T1\_A6) Außer den Grundrechenarten kennt Matlab natürlich auch komplexere mathematische Operationen. Testen Sie 2^3 und b^4. Was bedeutet dieser Haken?

T1\_A7) Ein weiteres Beispiel ist die Quadratwurzel - da es für diese aber kein Symbol auf der Tastatur gibt, ruft man sie in Form der Funktion sqrt auf. Dabei muss das Argument, die Zahl auf die die Funktion angewandt werden soll, in Klammern hinter den Funktionsnamen geschrieben werden: sqrt(16) oder d=sqrt(a). Weitere Funktionen sind z.B. sin, cos, exp, log, abs. Was könnten diese Namen bedeuten? Probieren Sie Ihre Vermutung mit Beispielen aus.

T1\_A8) Um nachzusehen, ob Sie richtig getippt haben, schauen Sie die Funktionsnamen in der Hilfe nach. Eine ganz kurze Beschreibung bekommen Sie, wenn Sie z.B. help sin eintippen. Sehr viel mehr Information gibt es im Hilfefenster: suchen Sie nach sin und den anderen Funktionsnamen.

T1\_A9) Manche mathematischen Operationen brauchen mehrere Argumente. Z.B. bei b/a kommt uns das ganz natürlich vor. Wenn es nicht wie bei den Grundrechenarten eine abkürzende Schreibweise gibt, werden Matlab-Funktionen alle Argumente durch Komma getrennt in der Klammer übergeben, die dem Funktionsnamen folgt. Z.B. wird der bei der ganzzahligen Division übrigbleibende Rest von der Funktion rem (remainder) berechnet, die zwei Argumente übergeben bekommt, z.B. rem(104, 10). Dabei ist die Reihenfolge der Argumente wichtig: was passiert, wenn man rem(10, 104) eingibt?

T1\_A10) Vielleicht haben Sie sich während der Übung bereits irgendwann vertippt. Was ist dann passiert? Falls Ihnen noch nichts dergleichen passiert ist, tippen Sie einmal ein e1=sin(2\*pi), dann e2=sine(2\*pi) und schließlich e3=sin(2pi). Wie reagiert Matlab darauf jeweils? Wie unterscheidet sich die Reaktion, wenn Sie f=10/0 eintippen?

**B) Vektoren und Matrizen:**

Vektoren und Matrizen sind das wichtigste Konzept der Programmierung in Matlab. Nahezu alle Daten werden in Matrizen organisiert und verarbeitet. Wenn Sie nicht sowieso mit Matrizenrechnung vertraut sind, lesen Sie bitte vor dem Kurs die kurze Einführung aus dem Buch "Matlab und Mathematik kompetent Einsetzen" von S. Adam.

Matrizen sind Verbundvariablen, die zusammengehörige Daten in einem Objekt vereinigen, indem Sie sie in einem Rechteckschema aus Zeilen und Spalten anordnen. Vektoren sind spezielle Matrizen, die entweder nur aus einer Zeile oder nur aus einer Spalte bestehen.

Der Doppelpunkt dient in Matlab dazu, Zahlenreihen zu erzeugen. Das ist insbesondere beim Zählen und beim Indizieren von großer Bedeutung.

Aufgaben:

T1\_B1) Rechnen Sie zunächst von Hand: a=[1; 2] (Spaltenvektor), b=[1 2] (Zeilenvektor). Was ist c=a\*b ? Multiplizieren Sie das Ergebnis c mit der Matrix d=[0 1; 0.1 10]. Überprüfen Sie Ihre Ergebnisse anschließend mit Matlab.

T1\_B2) Definieren Sie sich folgende Vektoren: a=[1 2 3], b=[0.1 0.2 0.3], c=[10; 20; 30] d=[100; 200], e=[-1; -2] und f=7.5. Welche Vektoren sind Spalten- und welche Zeilenvektoren?

Überlegen Sie und probieren Sie aus: Welche der Vektoren kann man addieren? Subtrahieren? Multiplizieren? Punktweise multiplizieren?

In welchen Fällen ist die Reihenfolge der Vektoren wichtig, in welchen nicht?

T1\_B3) Wenn Sie sich in den Kopf setzen, dass Sie gerne einen Vektor g definieren wollen, dessen Werte die Summe aus den Elementen von a und c sind, müssen Sie etwas tricksen: Einer der Vektoren muss transponiert werden. Probieren Sie aus: g1=a+c' und g2=a'+c. Was ist der Unterschied?

T1\_B4) Für den Umgang mit Vektoren werden von Matlab ein paar sehr praktische Funktionen bereitgehalten. Probieren Sie an ein paar Beispielen die Arbeitsweise folgender Befehle aus und sehen Sie im Zweifelsfall in der Hilfe nach:

sum(vektor), mean(vektor)

\*) Probieren Sie aus, welche der in A7) eingeführten Funktionen auch für Vektoren funktionieren.

T1\_B5) Definieren Sie sich zwei Matrizen: M1=[1 1 1; 5 6 7] und M2=[0 -1; -2 0.5; 0.1 1] und schauen Sie im Array Editor an. Welche Indizes hat der Eintrag 6 in M1? Greifen Sie einen Eintrag aus M2 heraus, z.B. M2(1,2)

T1\_B6) Achtung! Matlab kennt zwei Arten, eine Matrixkomponente zu indizieren: entweder, man benutzt die Schreibweise M1(1,2) mit (zeile, spalte) in der Klammer, oder man zählt die Matrix als einen langen Vektor durch, wobei die Spalten zu einem langen Spaltenvektor aneinandergehängt werden, z.B. M1(2). Welcher Einzelindex entspricht M1(1,2)?

T1\_B7) Berechnen Sie M1\*M2 und M2\*M1. Transponieren Sie M2 und berechnen Sie die Summe mit M1.

\* T1\_B8) Welche der Vektoren aus Aufgabe B2 kann man mit M1 oder M2 multiplizieren? Kann man sie auch addieren?

T1\_B9) Zum Glück braucht man in Matlab große Matrizen nicht komplett einzutippen. Probieren Sie aus, was passiert, wenn man o=ones(1,8) und z=zeros(3,3) eingibt.

T1\_B10) Eckige Klammern umschließen jeweils alle Komponenten einer Matrix oder eines Vektors. Man kann sie deshalb auch nutzen, um Vektoren oder Matrizen zu kombinieren. Überlegen Sie, was folgende Ausdrücke basierend auf den oben definierten Vektoren und Matrizen liefern sollten und probieren sie Ihre Hypothesen aus:

t1=[a b], t2=[a;b], t3=[a b f], t4=[d e], t5=[d;e],

t6=[a c], t7=[M1; a], t8=[M1; M2'], t9=[M1' M2];

t10=[[pi pi/2];[1 2];e']

T1\_B11) Der Doppelpunktoperator spielt in Matlab eine wichtige Rolle. In seiner einfachsten Form benutzt man ihn zum zählen: zahlen=1:10 erzeugt einen Vektor mit den Zahlen von 1 bis 10.

T1\_B12) "Zählen" kann man nicht nur in Schritten der Länge 1. Die Schrittweite wird zwischen Anfangs und Endwert in Doppelpunkte eingeschlossen angegeben. Probieren Sie aus: zahlen2= 10:10:100, zahlen3=0:0.1:0.5. Erzeugen Sie einen Vektor, der vom Startwert 5 mit Schrittlänge von 2.5 bis zum Endwert 20 läuft.

T1\_B13) Besonders wichtig ist der Doppelpunktoperator, beim Indizieren - also um auf definierte Elemente von Vektoren oder Matrizen zuzugreifen. Welche Werte ergibt zahlen(2:4)? Was könnte man alternativ schreiben, um an diese Werte heranzukommen? Greifen sie das 4. bis 7. Element aus dem Vektor zahlen2 heraus.

T1\_B14) Wenn der Doppelpunktoperator beim Indizieren alleine steht, z.B. zahlen(:) bedeutet es, dass der ganze Vektor genommen werden soll. Für Vektoren ist das recht langweilig, denn in diesem Fall könnte man auch einfach zahlen schreiben.

Anders ist das bei Matrizen. Definieren sie sich M3=[1 1 1 1; 5 6 7 8; 10 20 30 40]. Probieren Sie aus: M3(1,:), M3(:,3). Was bedeutet M3(:) im Gegensatz zu M3 und zu M3(:,:)?

T1\_B15) Und jetzt kombinieren wir das Gelernte: Greifen Sie aus M3 den dritten und vierten Eintrag der zweiten Zeile heraus.

T1\_B16) Eine sehr praktische Vereinfachung beim Indizieren ist der Begriff end. Beispielsweise der Ausdruck a(7:end) bezeichnet alle vom siebten bis zum letzten Eintrag des Vektors a. Dadurch braucht man die Länge des Vektors nicht zu kennen. Greifen sie die letzten drei Elemente aus dem Vektor zahlen heraus, indem Sie end benutzen.

T1\_B17) Noch sind Ihre Variablen so klein, dass Sie diese im Workspace-Fenster komplett angezeigt bekommen. Das wird sich aber bald ändern. Dann ist es sinnvoll, einen Befehl zu kennen, der einem die Größe von Variablen zurückliefert. Für einen Vektor v ist s=length(v) die Länge des Vektors. Bei Matrizen bietet es sich an, den Befehl size zu verwenden. Möchte man die Anzahl der Zeilen (also die Länge der 1. Dimension) wissen, sagt man zeilen=size(M3,1); die Spalten (2. Dimension) bekommt man mit spalten=size(M3,2).

**C) Grafische Darstellung von Vektoren:**

Die grafische Darstellung von Daten ist für viele Anwendungen höchst wichtig, gerade im wissenschaftlichen Bereich. Daten, die man einmal "gesehen" hat, kann man sich besser vorstellen, als wenn sie in Form von Vektoren vorliegen. Deshalb werden z.B. in Praktikumsprotokollen, Abschlussarbeiten etc. grundsätzlich grafische Darstellungen der Auswertungsergebnisse gefordert. Mit Matlab hat man die Möglichkeit, sich in Vektoren oder Matrizen angeordnete Daten sehr einfach grafisch darstellen zu lassen. Der Grundbefehl dafür lautet plot.

T1\_C1) Jetzt stellen wir einen Vektor grafisch dar: Definieren Sie sich den Vektor v=[1 1 1 1 1 2 2 2 2 2 3 3 3 3 3 0 0 0 0 0 1.5 1.5 1.5 1.5 1.5]. Lassen Sie sich diese Daten mit plot(v) grafisch darstellen. Was ist auf der x- und der y-Achse dargestellt?

T1\_C2) Erzeugen Sie sich mit Hilfe des Doppelpunktoperators einen Vektor mit einer Zahlenreihe, z.B. v=0.01:0.01:10; und stellen diesen mit plot(v) grafisch dar. Probieren Sie verschiedene Zahlenreihen aus. Was passiert mit den Achsen?

\*T1\_C3) Erzeugen Sie sich eine Matrix M, die aus mehreren verschiedenen Zahlenreihen besteht und stellen diese mit plot(M) dar. Was ist zu sehen? Wie kann man eine einzelne Zeile der Matrix grafisch darstellen? Wie eine Spalte?

T1\_C4) Bisher wurden die Werte jeweils gegen den index der Vektoren bzw Matrix aufgetragen. Das ist nicht immer sinnvoll. Generieren Sie sich zwei Vektoren x=-1:0.1:1; und y=x.^2; mit plot(x,y) werden die Werte gegeneinander aufgetragen.

Was passiert bei plot(y,x)?

Erzeugen Sie Glockenkurven für verschiedene Vektoren x.

T1\_C5) Um sich nur einen Ausschnitt der vorhandenen Daten anzusehen, oder um Daten in Abbildungen zu skalieren gibt es den Befehl axis([xmin xmax ymin ymax]). Probieren Sie verschiedene Achsenskalierungen für Ihre Glockenkurven aus.

Erzeugen Sie sich außerdem eine Glockenkurve mit einem größeren Definitionsbereich, z.B. von -10 bis 10, plotten Sie diese und sehen sie sich mit verschiedenen Skalierungen an. Wie bekommt man einen plot der genauso aussieht wie der ursprüngliche in T1\_C4?

T1\_C6) Es gibt eine Reihe von Optionen, um Daten möglichst schön oder praktisch aufzutragen. Probieren Sie plot(y,x,'r') aus. Was bewirken die folgenden Optionen: r, b, g, k, c, m, \*, ^, s, d, :, -, --

Notieren Sie sich die jeweilige Bedeutung und probieren Sie sinnvolle Kombinationen aus, z.B. plot(y,x,'r^:')

**D) Hausaufgaben:**

T1\_H1) Der Mathematiker Karl Friedrich Gauß sollte als Grundschüler damit beschäftigt werden, alle Zahlen von 1 bis 100 zusammenzuzählen. Er war innerhalb weniger Minuten damit fertig, da er das Prinzip erkannte, dass jede Summe einer Zahl aus der oberen Hälfe der Reihe mit einer passenden Zahl aus der unteren Hälfte den Wert 101 ergab (z.B. 1+100=101, 2+99=101 etc). Da es 50 solcher Paare gibt, ist die Lösung 5050=101\*50.

Die allgemeine Formel für die Summe einer Reihe natürlicher Zahlen von a bis b lautet

s=1/2\*(a+b)\*(b-a+1).

Testen Sie diese Formel, indem sie verschiedene Reihen mit dem Befehl r=a:b erzeugen und deren Summe mit sum(r), sowie mit der obigen Formel berechnen.

T1\_H2) Erzeugen Sie eine Matrix mit einer Einmaleins-Tabelle für das kleine Einmaleins, also

 1 2 3 ... 10

 2 4 6 ...

 3 6 9 ...

 4 8 12 ...

 ... ... ...

 10 20 30 100

(natürlich vollständig ausgefüllt).

T1\_H3) Verwenden Sie das Prinzip, Matrizen als Bestandteile von Matrizen zu definieren, um ein 8\*8 Schachbrett aus 0 und 1 zu erzeugen.

Sehen Sie sich Ihr Resultat im Array Editor an.

Man kann sich den Inhalt einer Matrix auch grafisch anzeigen lassen. Wenn Ihre Matrix-Variable "schach" heißt, tippen Sie einfach ein imagesc(schach).

T1\_H4) Durch welche Abfolge von Matlab-Befehlen erzeugt man den Inhalt (ohne Beschriftung) der folgenden Tabelle der Höhendifferenzen:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Duforspitze  | Matterhorn | Gornergrat | Zermatt | Brig |
| Duforspitze  | 0 | -156 | -814 | -3014 | -3956 |
| Matterhorn | 156 | 0 | -658 | -2858 | -3800 |
| Gornergrat | 814 | 658 | 0 | -2200 | -3142 |
| Zermatt | 3014 | 2858 | 2200 | 0 | -942 |
| Brig | 3956 | 3800 | 3142 | 942 | 0 |

aus dem Vektor der Höhenwerte v=[4634; 4478; 3820; 1620; 678] ?

\*T1\_H5) Wer Grafik noch schöner haben möchte als es mit den in T1\_C6 erwähnten Grafikoptionen möglich ist, kann dafür sogenannte "line properties" nutzen. Diese werden der Plotfunktion jeweils als Schlüsselbegriff und Wert übergeben, z.B. plot(x,y,'LineWidth',5). Weitere line properties sind 'Color', 'MarkerFaceColor', 'MarkerEdgeColor', und 'MarkerSize'. Was bedeuten diese? Stellen Sie Ihre Glockenkurve mit einer dicken roten Linie und blauen Dreiecken als Datenpunkten dar.

\*T1\_H6) Spielen Sie ein wenig mit den interaktiven Möglichkeiten, Ihre Grafik zu verändern: zoomen sie auf einen gewissen Bereich, ändern Sie die Liniendicke, geben Sie der Grafik einen Titel etc. Was für Möglichkeiten gibt es, die Grafik interaktiv zu verschönern?