



Einführung in Matlab

1. Grundlagen

Prof. Dr. Christiane Zarfl, Dipl.-Inf. Willi Kappler, Prof. Dr. Olaf
Cirpka



- In diesen Kursunterlagen werden die folgenden Konventionen verwendet:
- Eingabetext wird grün hinterlegt, z.B.: `plot`
- Alle Ausgaben werden blau hinterlegt, z.B.: `a = 12.78`
- Beispielprogramme werden grün hinterlegt mit Zeilennummern abgebildet und die Ausgabe dieser Programme wird blau hinterlegt, z.B.:

```
1 1.5 + 2.75
2 3.12 * 7.87
3 20.97 / 2.10
4 sin(pi / 2.0)
```

```
1 ans = 4.2500
2 ans = 24.554
3 ans = 9.9857
4 ans = 1
```



- Hyperlinks auf Webseiten werden mit blauem Text gekennzeichnet, z.B.: [Uni Tübingen - klick mich](#)
- Menüeinträge werden durch hellgraue Kästen dargestellt, z.B.: Datei Speichern
- Tastenkürzel (Shortcuts) und Eingaben werden ebenfalls in hellgrauen Kästen dargestellt, z.B.: F10, ↵, ctrl-a
- Übungsaufgaben werden in einem gelben Kasten abgebildet, z.B.:

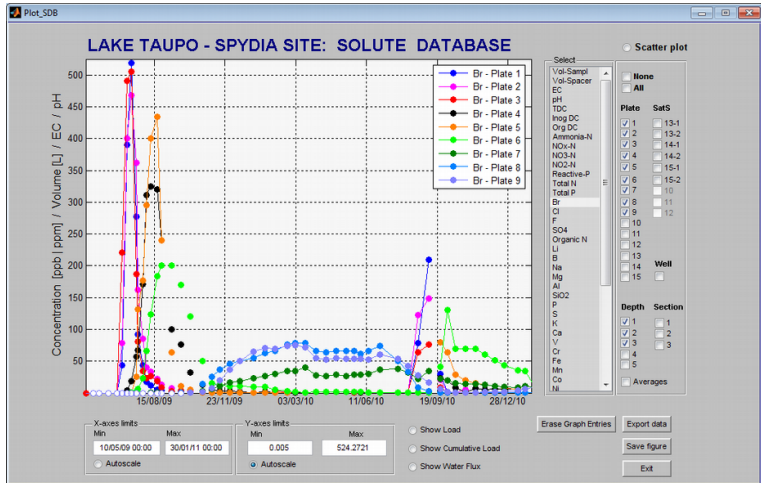
Übung 1: Berechnen Sie die ersten 20 Dezimalziffern der Zahl π .



- Einfache Rechnungen
- Datenaufbereitung & -speicherung
- Visualisierung von Daten & Simulationsergebnissen
- einfaches Lösen von (linearen) Gleichungssystemen, Differentialgleichungen (DGL), DGL-Systemen...
- Wiederverwendung von häufig gebrauchten Berechnungen (Programmierung)
- Datenanalyse (z.B. Regression) & Statistik
- Komplexe Modellierung von Umweltsystemen

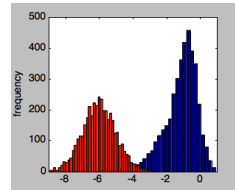
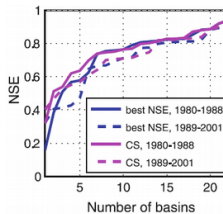
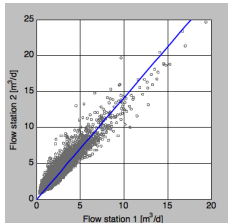
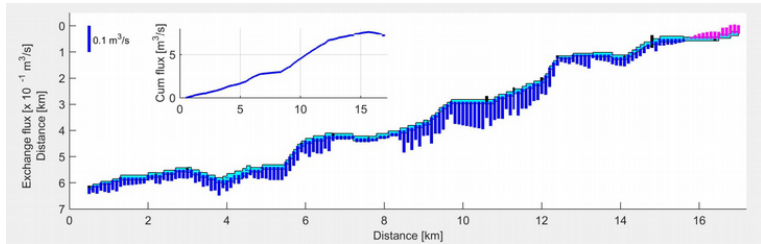


Visualisierung von Daten & Simulationsergebnissen



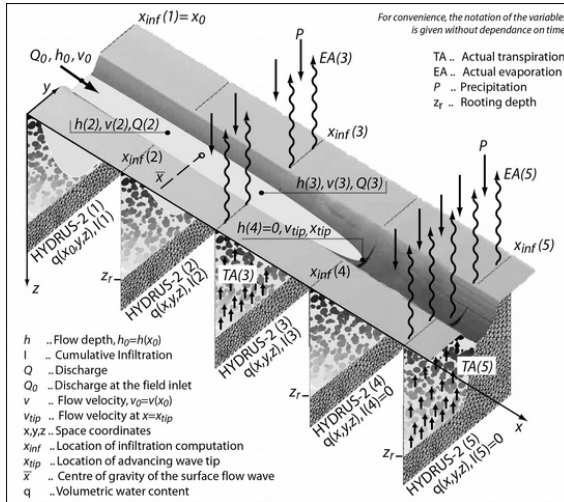


Datenanalyse & Statistik





Modellierung von Umweltsystemen





- Hardware:
 - BYOD (eigenes Gerät mitbringen)
 - Geo-Notebooks (Raum S245)
 - CIP Pool Rechner (Raum S310)
- Software:
 - Auf Institutshardware bereits vorinstalliert
 - ZDV: Matlab [herunterladen](#)
 - [GNU Octave](#) (Open Source, aber nicht 100% kompatibel)
- Auf Institutshardware bitte zuerst ein eigenes Verzeichnis anlegen!



- “*Matlab* ist eine kommerzielle Software des Unternehmens *The MathWorks, Inc.* zur Lösung mathematischer Probleme und zur grafischen Darstellung der Ergebnisse.” (Quelle: [Wikipedia](#)).
- Matlab leitet sich ab von **MAT**rix **LAB**oratory.
- Wir benutzen Matlab als (numerische) Programmiersprache.
- Wie ein Taschenrechner oder Excel arbeitet Matlab numerisch (mit Zahlenwerten, also nicht symbolisch wie ein [CAS](#)).
- Anders als bei einem Taschenrechner können Zahlenwerten Variablennamen zugewiesen werden.
- Im Programm werden die Variablennamen als Platzhalter für die Werte verwendet.



- kennen Sie den Aufbau der Oberfläche der Software Matlab.
- benutzen Sie die Matlab-Hilfe, um für Sie nützliche Funktionen und Informationen selbstständig zu finden.
- führen Sie einfache Rechnungen mit Matlab durch.
- können Sie Variablen in Matlab definieren und verwenden.
- kennen Sie die Vorteile der Verwendung von Vektoren und können diese in Matlab definieren und für Rechnungen verwenden.



gegenwärtiges Verzeichnis

The screenshot shows the MATLAB R2014a environment. The 'Current Folder' pane on the left lists files: 'Hausaufgaben', 'Woche1', 'Woche2', 'Woche3', and 'template.m'. An arrow points from the text 'Dateien im gegenwärtigen Verzeichnis' to this pane. The 'Editor' pane in the center shows the 'template.m' script. A yellow highlight covers the variable declarations: `x = 0:0.1:5;`, `y1 = sin(x);`, `y2 = sin(3*x);`, and `y3 = sin(0.6*x);`. An arrow points from the text 'Variablen im Arbeitsspeicher' to this highlighted section. The 'Workspace' pane on the right shows the variables: `x` (1x51 double, 0 to 5), `y1` (1x51 double, -0. to 0.9), `y2` (1x51 double, -0. to 0.9), and `y3` (1x51 double, 0 to 0.9). An arrow points from the text 'Eingabeverlauf' to this pane. The 'Command Window' at the bottom shows the prompt `>>` and the command `template`. An arrow points from the text 'Eingabefenster mit "Prompt" (>>)' to this window.


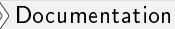

↑
Dateien im
gegenwärtigen
Verzeichnis

↑
Variablen im
Arbeitsspeicher


↑
Eingabefenster mit
"Prompt" (>>)

↑
Eingabe-
verlauf



- Niemand kann alle Befehle kennen, deshalb ist die (ausführliche) Hilfe in Matlab so wichtig.
- Allgemeines Hilfe-Fenster:   ()
- Information zu einem Befehl:
 - `doc <Befehlsname>` (Info in einem extra Fenster)
 - `help <Befehlsname>` (Info im Befehlsfenster)
- Beispiele: Im Prompt eingeben:
 - `help sin`
 - `help exp`
 - `doc plot`



- Alle Anweisungen werden nach dem Prompt (`>>`) eingegeben und mit  (Return) bestätigt. Matlab nennt das Ergebnis `ans` (für answer):

```
1 12/3 + 7*5 - 1
```

```
1 ans = 38
```

- Addition (\oplus), Subtraktion (\ominus), Multiplikation (\otimes) und Division (\oslash) wie im Taschenrechner (Matlab kennt “Punkt vor Strich-Rechnung”).
- Braucht man einen **Ausdruck öfters**, so kann man ihn als **Variable** definieren:

```
1 a = 48/3 - 3^2
```

```
1 a = 7
```

- Variablen werden im Arbeitsspeicher (Workspace) gespeichert (s. Arbeitsspeicher-Fenster).



- Ein Semikolon (;) am Ende der Eingabezeile unterdrückt die Ausgabe des Ergebnisses.
- Matlab **unterscheidet** zwischen Groß- und Kleinbuchstaben!
- Potenzieren wird **vor** einer **Multiplikation** oder **Division** ausgewertet, sonst gilt “Punkt-vor-Strich”; runde Klammern “(” und “)” um die Reihenfolge der Berechnung zu steuern.
- Mehrere Anweisungen in einer Zeile sind zulässig:
 - Sind sie durch ein Komma getrennt, so folgt eine Ausgabe.
 - Werden sie durch ein Semikolon getrennt, so folgt keine Ausgabe:

```
1 b = (3+5) * 6;  
2 c = (b/3) ^ 2, d = 1/c ^ 2
```

```
1 c = 256  
2 d = 1.5259e-05
```




- Regeln bei der Definition von Variablen:
- Das erste Zeichen muss ein Buchstabe sein (Keine Zahl!)
- Keine Sonderzeichen (außer Unterstrich)
- Max. Zeichenlänge (abhängig vom Computer)
- **Vorsicht:** Variablennamen identisch mit Funktionen ist erlaubt, hat aber Seiteneffekte!
- Variablen haben einen bestimmten **Typ**, z.B. Ganzzahl, Fließkommazahl, Vektor, Matrix, ...





- `who`: gibt eine Liste der Variablen im Arbeitsspeicher aus
- `whos`: gibt zusätzliche Information (Typ, Größe, Speicherbedarf)
- `clear <Variable>`: löscht die Variable
- `clear all`: löscht alle Variablen
- `clc`: löscht den Inhalt des Befehlsfensters




Geben Sie nacheinander folgende Anweisungen ein. Überlegen Sie vorher, was Matlab ausgegeben wird?

`u = 2, v = 5;` 

`(u+6) / 4` 

`y = x+1` 

`y = 3u` 

Welche der folgenden Variablennamen sind **nicht** zulässig?

`anzahl`, `Summe_a+b`, `5_Tageskarte`, `dauer_phase3`, `sin`



- Es gibt in Matlab bereits “eingebaute” Funktionen (viel mehr als im Taschenrechner und Excel).
- Z.B. die Wurzelfunktion (square root): `a = sqrt(2)/2`
- Nähere Informationen zur Funktion `sqrt` erhält man mit `help sqrt`.
- Funktionen können keinen, einen oder mehrere **Eingabeparameter** haben.
- Funktionen können keinen, einen oder mehrere **Rückgabewerte** haben.
- Wie bei Variablen auch haben Funktionen einen bestimmten Typ. D.h. die Ein- und Ausgabewerte **müssen** vom Typ her passen.



Berechnen Sie den natürlichen Logarithmus von 1.36

Berechnen Sie auch den Logarithmus zur Basis 10 von 1.36

(Zusatz: Wie berechnen Sie den Logarithmus zur Basis 3 von 1.36 ?)



Berechnen Sie $\cos(\pi)$ und $\cos(\pi/2)$

π ist in Matlab bereits eingebaut und wird mit `pi` bezeichnet

Vorsicht: Das Argument der trigonometrischen Funktionen (`sin`, `cos`, `tan`, `cot`) wird von Matlab immer im **Bogenmaß** interpretiert

Berechnen Sie den Kosinus von 180° und 90°



- Eine Gruppierung von mehreren Zahlenwerten nennt man einen **Vektor**.
- Eine zweidimensionale Gruppierung von Zahlen nennt man eine **Matrix**.
- **Es folgt:** Eine Zahl ist sowohl ein spezieller Vektor (der Länge 1), als auch eine spezielle Matrix der Dimension 1×1 .
- **Ein Vektor der Länge n** ist eine spezielle Matrix der Dimension $n \times 1$ (Spaltenvektor) oder $1 \times n$ (Zeilenvektor).
- Matlab (**MAT**rix **LAB**oratory) kennt intern nur Matrizen!
- Die Berechnung der Wurzel in Matlab von vorhin:

```
a = sqrt(2)/2
```

Hier ist a eine 1×1 Matrix (selbst die Konstante 2 wird als eine konstante 1×1 Matrix interpretiert).



- Vektoren werden in Matlab immer in eckigen Klammern eingegeben: “[” und “]”
- Die Elemente des Vektors sind durch Leerzeichen (oder ein Komma) zu trennen (es ergibt sich ein Zeilenvektor, Semikolon als Trenner ergibt einen Spaltenvektor).
- Um in Matlab Tipparbeit zu sparen können Vektoren verwendet werden. Möchte man z.B. die Quadratwurzel mehrerer Werte berechnen, so geht man folgendermaßen vor:
- Zeilenvektor in einer Variablen definieren: `x = [0 2 4 6 8 10]`.
- Die Quadratwurzelfunktion auf den gerade definierten Zeilenvektor `x` anwenden: `y = sqrt(x)`.
- Das Ergebnis in der Variablen `y` ist nun ebenfalls ein Zeilenvektor und enthält die einzelnen Quadratwurzeln der oben aufgeführten Zahlen.



- Auf die Elemente eines Vektors wird mit der runden Klammer zugegriffen “()”. Z.B. `x(3)` gibt das dritte Element von `x` zurück.
- Mit dem Doppelpunkt-Operator `:` (auch `colon` genannt) können in Matlab Zahlenfolgen als Vektoren definiert werden. Z.B. kann man `x = [0 2 4 6 8 10 12]` kürzer schreiben als `x = 0:2:12`.
- Die generelle Syntax lautet:
`Startwert:Schrittweite:Endwert` oder
`Startwert:Endwert` mit Schrittweite `1`.
- Schrittweiten dürfen auch negativ sein: `u = 29:-2:0`



Erzeugen Sie einen Vektor y , der die Funktionswerte des natürlichen Logarithmus an den Stellen $x = 1, 3, 5, 7, 9$ enthält.

Was gibt Matlab aus, wenn Sie $y(1)$ eingeben?



Geben Sie die Vektoren a und b mit den Elementen $-10, -8, -6, \dots, 6, 8, 10$ bzw. $10, 9, 8, \dots, 0$ mit kurzen Anweisungen ein. **Tipp:** `help colon`

Sei $x = [1 \ 3 \ -2 \ 6 \ 0 \ 7 \ 11 \ -8 \ -5]$. Finden Sie mit `help paren` und `help colon` heraus, wie man aus x einen Vektor bildet, der

- das 1., 4. und 9. Element von x enthält
- aus den ersten 4 Elementen von x besteht
- jedes zweite Element von x enthält.



- Vektoraddition (-subtraktion) und die Multiplikation (Division) eines Vektors mit einem Skalar sind elementweise definiert durch:
- $w = [1 \ 2 \ 3] * 2$: Jedes Element des Vektors $[1 \ 2 \ 3]$ wird mit 2 multipliziert und in der Variable w gespeichert.
- $[2 \ -1 \ 9] + [1 \ 3 \ 6]$ ergibt den Vektor $[3 \ 2 \ 15]$
- **Achtung:** Es können nur Vektoren (Matrizen) mit gleichen Dimensionen addiert oder subtrahiert werden!
- Die elementweise Addition (Subtraktion) eines Vektors mit einem Skalar ist zulässig, z.B.: $[2 \ 5 \ 7] + 3$, ergibt $[5 \ 8 \ 10]$



- Elementweise Multiplikation, Division und Potenzieren bedarf einem extra Operator in Matlab ("Punktoperationen")
- Gegeben seien `a = 1:2:10; b = 1:5;`
- Elementweise Multiplikation von a und b: `a.*b`, ergibt hier:
`[1 6 15 28 45]`
- Elementweise Division: `a./b`
- Elementweises Potenzieren: `a.^2`
- **Vorsicht:**
 - `a` und `b` müssen gleiche Dimension haben!
 - `a.*b` und `a*b` sind ein großer Unterschied!



- Der Stern (Asterisk, `*`) in Matlab definiert die übliche Vektormultiplikation.
- `a*b` ergibt somit einen Fehler, weil die Dimensionen von `a` und `b` 5×1 ist und diese nicht sinnvoll als Vektoren multipliziert werden können.
- Das Apostroph `'` transponiert einen Vektor (oder eine Matrix), `a'` ist also ein Vektor der Dimension 1×5 .
- Mit diesem Wissen zurück zur Multiplikation:
- `a*b'` ergibt das Skalarprodukt von `a` und `b`.
- `a'*b`, ergibt das dyadische Produkt (5×5 Matrix).



Gegeben seien $a = [1 \ 4 \ 6]$ und $b = [-1 \ 2 \ 1]$.

Was gibt Matlab aus, wenn Sie die folgenden Anweisungen eingeben (geben Sie die Antwort, **bevor** Sie mit Matlab rechnen!):

$a+b$

$a*2, a/2$

$a+3$

$a*b$

$a.*b$

$a./b$

$a*b'$

$a'*b$



Sie möchten einen Vektor x mit den Elementen $0, 0.2\pi, 0.4\pi, \dots, 10\pi$ definieren. Dazu können Sie zum Beispiel die elementweise Multiplikation mit π verwenden. Welche Möglichkeiten gibt es?

Wie würde Matlab die Anweisung `y = [0:0.2:10*pi]` interpretieren?



- Vorteile:
 - Matlab kennt eine große Anzahl an nützlichen Funktionen und Algorithmen (siehe `doc`)
 - Rechnen mit Variablen als Platzhalter ermöglicht einfache Darstellung (Lesbarkeit)
 - ...
- Fallstricke:
 - Sequentiell arbeitendes Programm, d.h. Matlab arbeitet Zeile für Zeile.
 - Eine Variable muss erst definiert werden, bevor man sie benutzen kann.
 - Ändert man den Wert einer Variable, müssen alle Operationen mit dieser Variable wiederholt werden.
 - Matlab ist “amerikanisch”
 - Dezimalpunkt statt -komma
 - (Standard-)Funktionen haben englische Namen (z.B. “`sqrt`”)
 - ...



- Wie kann ich mehrere Rechenschritte/Matlab-Befehle, die ich immer wieder benötige, “speichern” und zusammenfassen?
- Wie kann ich Ergebnisse von Berechnungen grafisch darstellen?