Einführung in MATLAB

Technische Programmiersprache mit

- einfacher Synatx
- grosser Anzahl zur Verfügung stehender math. Funktionen
- Graphikausgabe

Variablen und Wertzuweisung

- **Variablen** müssen nicht explizit ausgewiesen werden
  ⟶ keine *int, double, boolean* Festlegung notwendig
- **Wertzuweisung** im Programm:

```
x=2.5;
N=100;
l=0;      % FALSE
```

  oder durch Einlesen von der Tastatur:

```
x=input('Geben Sie x ein:  ');
```

- Abfrage nach existierenden Variablen: `who`
- Löschen alter/aller Variablenbelegungen: `clear;`

Ausgabe von Ergebnissen

- **Numerische Ausgabe** durch:

  `N` (Bildschrim)
  `Ergebnis=x` (innerhalb Programm)

  ⟶ d.h. Weglassen des Semikolons führt zur Ausgabe der Operation

  formatierte Ausgabe:

```
fprintf('Ergebnis:  %5.0f\n',x);
```

- **Grafische Ausgabe:** später

Relationen und Verknüpfungen

- **Relationen**

$<$	kleiner
$<=$	kleiner gleich
$>$	größer
$>=$	größer gleich
$==$	gleich
$\sim=$	ungleich

- **Verknüpfungen**

&	und
|	oder
$\sim$	Negation

Wichtig für: Abbruchbedingung bei `if-else` bzw. `while`-Anweisungen

Arithm. Operationen und MATLAB-Funktionen

- **Arithmetische Operationen**:

+	Addition
−	Subtraktion
∗	Multiplikation
/	Division
^	Exponentiation

- **MATLAB-Funktionen**: `cos, sin, exp, ...`

  ```
  x=rand(1);
  ```

  erzeugt eine auf dem Intervall (0,1) gleichverteilte Zufallszahl

if-else Anweisung

```
if    (Vergleichsausdruck 1)
      (Anweisungsblock 1);
else if   (Vergleichsausdruck 2)
      (Anweisungsblock 2);
else
      (Anweisungsblock 3);
end
```


for-Schleife

Zählschleife mit vorher bekannter Länge

```
for i=1:N
      (Anweisungsblock);
end
```

Zählweise in umgekehrter Richtung:

```
for i=N:-1:1
```


while-Schleife

Schleife mit vorher unbekannter Länge

```
while (Vergleichsausdruck)
      (Anweisungsblock);
end
```

Vektoren

- Erzeugung **"einfacher" Vektoren**

```
a=1:10;
```

  erzeugt Zeilenvektor $a = (1\ 2\ \ldots\ 10)$.

- Eingabe von **Zeilenvektoren**

  `v=[1,4,8,9]` oder `v=[1 4 8 9]`

  erzeugt $v = (1\ 4\ 8\ 9)$

- Eingabe von **Spaltenvektoren**

  `v=[1,4,8,9]'` oder `v=[1 4 8 9]'` oder `v=[1;4;8;9]`

  erzeugt

$$v = \begin{pmatrix} 1 \\ 4 \\ 8 \\ 9 \end{pmatrix}$$

- Vektoren mit **"beliebigen" Einträgen**

```
for i=1:N
   b(i)= (Anweisung);
end
```

- **Zugriff** auf ein Element eines Vektors

```
v(2)
```

  liefert als Ergebnis das zweite Element des Vektors. Die Indizierung der Elemente beginnt bei 1.

Grafiken

Vektoren werden in MATLAB u.a. bei der Erstellung von Grafiken verwendet:

```
plot(a,b);
```

erstellt einen xy-Plot der Werte aus den Vektoren a (x-Achse) und b (y-Achse).
!!! Die Vektoren a und b müssen die gleiche Anzahl an Elementen haben !!!

Grafikoptionen

- **Benennung** der x- und y- Achse:

```
xlabel('nameX');
ylabel('nameY');
```

- **Skalierung** der x- und y-Achse:

```
axis([xmin xmax ymin ymax]);
```

- **Linienzug**

```
plot(a,b,'--');
```


Weitere Befehle

`clock` aktuelle Uhrzeit
(Dezimalform: Jahr, Monat, Tag, Std, Min, Sek)

`etime(t1,t2)`
Differenz in Sekunden zwischen Zeit t1 und t2

`\n` Zeilenumbruch
z.B. `fprintf('Zeilenende\n');`

`floor(x)`
größte Integer-Zahl, die kleiner oder gleich x ist
z.B. `floor(3.5)=3, floor(-5.6)=-6`

`ceil(x)`
kleinste Integer-Zahl, die größer oder gleich (x) ist
z.B. `ceil(3.5)=4, ceil(-5.6)=-5`

`%` Kommentare in MATLAB-Programmen

Programme in MATLAB 1

- Suffix: `.m`, z.B. `test.m`
- Sollten im Arbeitsverzeichnis gespeichert werden (in dem auch MATLAB gestartet wird)
- Programmeingabe über Editor: `nedit program.m &`

Aufbau

als Routine

```
function summe
   (Funktionsblock);
```

mit Eingabeparameter

```
function summe(a,b)
   (Funktionsblock);
```


mit Ausgabe- und Eingabeparameter

```
function summeErgebnis=summe(a,b)
   (Funktionsblock);
```

Hinweis: Der Name der Funktion ist immer gleich dem Namen des Programmes (in unserem Fall heißt das Programm also `summe.m`). In jedem Programm darf es nur eine Funktion geben.

Programme in MATLAB 2

Aufruf in MATLAB

> `summe`

> `summe(5,8)` oder `summe(x,y)`

> `c=summe(5,8)` oder `c=summe(x,y)`

Iterative & Rekursive Programme 1

Problem:

$$S10 = 1 + 2 + \ldots + 10 = \sum_{i=1}^{10} i$$

Iterative Berechnung

Die Funktion `s10` wird nur einmal aufgerufen

```
function s10()
   s10=1+2+3+4+5+6+7+8+9+10;
   s10           % bewirkt Bildschirmausgabe
```

Aufruf in MATLAB: > s10

Iterative & Rekursive Programme 2

Rekursive Berechnung

Die Funktion ruft sich immer wieder selbst auf:

$$\begin{aligned} S10 &= 10 + S9 \\ S9 &= 9 + S8 \\ &\vdots \\ S2 &= 2 + S1 \\ S1 &= 1 \end{aligned}$$

D.h. für eine rekursive Definition benötigen wir einen Startwert.

```
function sR=sumRek(n)
   if (n==1)
      sR=1;
   else
      sR=n+sumRek(n-1);
   end
```

Aufruf in MATLAB: > sumRek(10)