

Mathematik am Computer

MATLAB, Teil I

M. Grote, H. Harbrecht und M. Schmidlin

Universität Basel

12.10 – 15.10.2026

Übersicht

1 Bedienung von Matlab

2 Skripte

3 Steuerung

Grundlegendes

Matlab Features:

- ① **Command Window** zum direkten Ausführen von Matlab-Befehlen
- ② **Command History** Liste ausgeführter Befehle. Erneutes Ausführen durch Anklicken
- ③ **Current Folder** Aktuelles Verzeichnis — hier werden selbstprogrammierte Befehle gesucht.
- ④ **Workspace** Liste aller aktuell belegten Variablen
- ⑤ **Editor** zum Schreiben von eigenen Befehlen

Befehle stets mit Return beenden.

Abbruch: `strg+C`.

Beispiel

MATLAB R2019b - academic use

HOME PLOTS APPS EDITOR PUBLISH VIEW

FILE NAVIGATE EDIT BREAKPOINTS RUN

Current Folder: C:\Users\SRJM\Dropbox\Doktorat Simon Michel\Lehre\HS19 Praktikum\Serie 04\Eigene Loesung

Editor - Untitled

```

1

```

Workspace

Name	Value	Size
ans	122	1x1
v	[2;3]	2x1
x	11	1x1

Command Window

```

>> v = [2;3]

v =

     2
     3

>> x = v(1) + 3*v(2)

x =

    11

>> x^2 + 1

ans =

    122

```

Command History

```

%-- 14.10.2019 16:10 --%
clear
close all
clc
v = [2;3]
x = v(1) + 3*v(2)
x^2 + 1

```

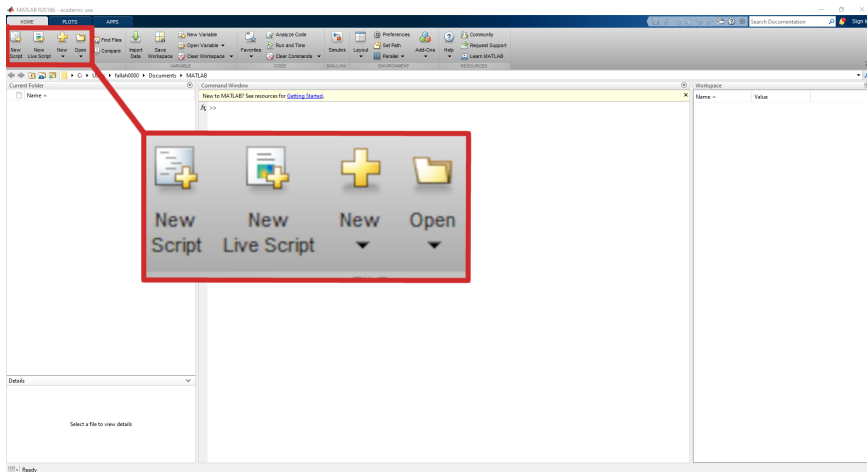
script | Ln 1 | Col 1

Matlab-Bedienung und Hilfe

Aufgrund der grossen Anzahl von Befehlen und weiterer Optionen gibt es Hilfestellungen um sich die volle Funktionalität und Syntax einzelner Befehle zu merken bzw. sich daran zu erinnern:

- ① `help` + Funktionsname gibt die Hilfe dazu aus
- ② Matlab-Hilfsmenü `doc`: Umfangreiche Suchmöglichkeiten
- ③ Buchstabenfolge + Tabulator: Automatische Befehlsergänzung
- ④ Pfeil-nach-oben: Befehlsverlauf
- ⑤ Online Ressourcen (`doc` ist auch über den Browser einsehbar, Foren, usw.)

Erstellen



Für die Abgabe der Aufgaben können sowohl MATLAB-Skripte wie -Live Skripte erstellt werden, z.B. `S04A02.m`, `S04A03.m`, `S04A04.m` bzw. `S04.mlx` usw.

Erläuterungen

Eine Datei, welche eine Abfolge von Befehlen enthält, heisst **Skript** oder Programm.

Name der Datei beliebig, Dateierweiterung muss `.m` sein.

Aufruf des Skripts (ohne Dateierweiterung) führt die im Skript enthaltenen Befehle sequenziell aus.

Semikolon `;` unterdrückt die Ausgabe.

Kommentare beginnen mit einem Prozentzeichen `%`.

Hinweis: Kommentare sind nützlich um einen Code verständlich zu halten, für andere oder auch für sich selbst, wenn der Code nach langer Zeit wieder benutzt werden muss.

Achtung: Ein MATLAB-Skript und ein MATLAB-Live Skript ist nicht ganz dasselbe!

Erläuterungen

Einschränkungen bei Namen von Variablen und Dateien

- alle Namen müssen sich unterscheiden

- Gross- und Kleinbuchstaben werden unterschieden

- Namen müssen mit einem Buchstaben beginnen

- Namen dürfen keine Sonderzeichen enthalten

Was ist eine Variable?

Was bedeutet " $x = x+1$ "?

Beispiel:

Eingabe	Zuweisung	
	x	y
$x = 3;$	3	
$y = 4;$	3	4
$x = x * y;$	12	4
$x = x - y;$	8	4
$y = x^2;$	8	64

if-elseif-else-end

```
if n == 1
    Befehle1
elseif n == 2
    Befehle2
    :
elseif n == N
    BefehleN
else
    BefehleAlt
end
```

Falls $n = 1$ ist, so werden *Befehle1* ausgeführt.

Falls $n = 1$ falsch ist und $n = 2$ gilt, so werden *Befehle2* ausgeführt.

Falls alle vorherigen Bedingungen falsch sind und $n = N$ gilt, werden *BefehleN* ausgeführt.

In allen anderen Fällen werden *BefehleAlt* ausgeführt.

Die for-Schleife

```
for k = 4:n  
    Befehle  
end
```

- ① Zunächst ist $k = 4$ und es werden alle Befehle zwischen `for` und `end` mit dem Wert $k = 4$ ausgeführt.
- ② Es wird $k = 5$ gesetzt und alle Befehle zwischen `for` und `end` mit dem Wert $k = 5$ ausgeführt, usw.
- ③ Es werden alle Werte von k durchlaufen, bis einschliesslich $k = n$.

Genauer: `for k = Vektor`
 k durchläuft den *Vektor* von Anfang bis Ende.

Die while-Schleife

```
while  $t > 0$   
    Befehle  
end
```

- ① Falls $t > 0$ gilt, so werden alle Befehle zwischen `while` und `end` ausgeführt.
- ② Es wird wieder geprüft, ob $t > 0$ gilt. Falls dies erfüllt ist, so werden wieder alle Befehle zwischen `while` und `end` ausgeführt.
- ③ Dies wiederholt sich solange bis $t > 0$ nicht erfüllt ist.

Die while-Schleife

Gilt $t > 0$ am Anfang nicht, so werden alle Befehle übersprungen.

Man muss t vor `while` $t > 0$ einen Wert zuweisen.

Gilt $t > 0$ immer, so bricht der Programmablauf niemals ab.
Deswegen wird der Wert von t normalerweise innerhalb der Schleife geändert.

Genauer: $t > 0$ liefert als Ergebnis entweder 0 (`false`) oder 1 (`true`). Die Schleife wird solange durchlaufen, bis der Wert 0 (`false`) ist.

Erlaubt ist auch z.B. `while (t>0) && (t<1)`.