UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

Introduction to Matlab

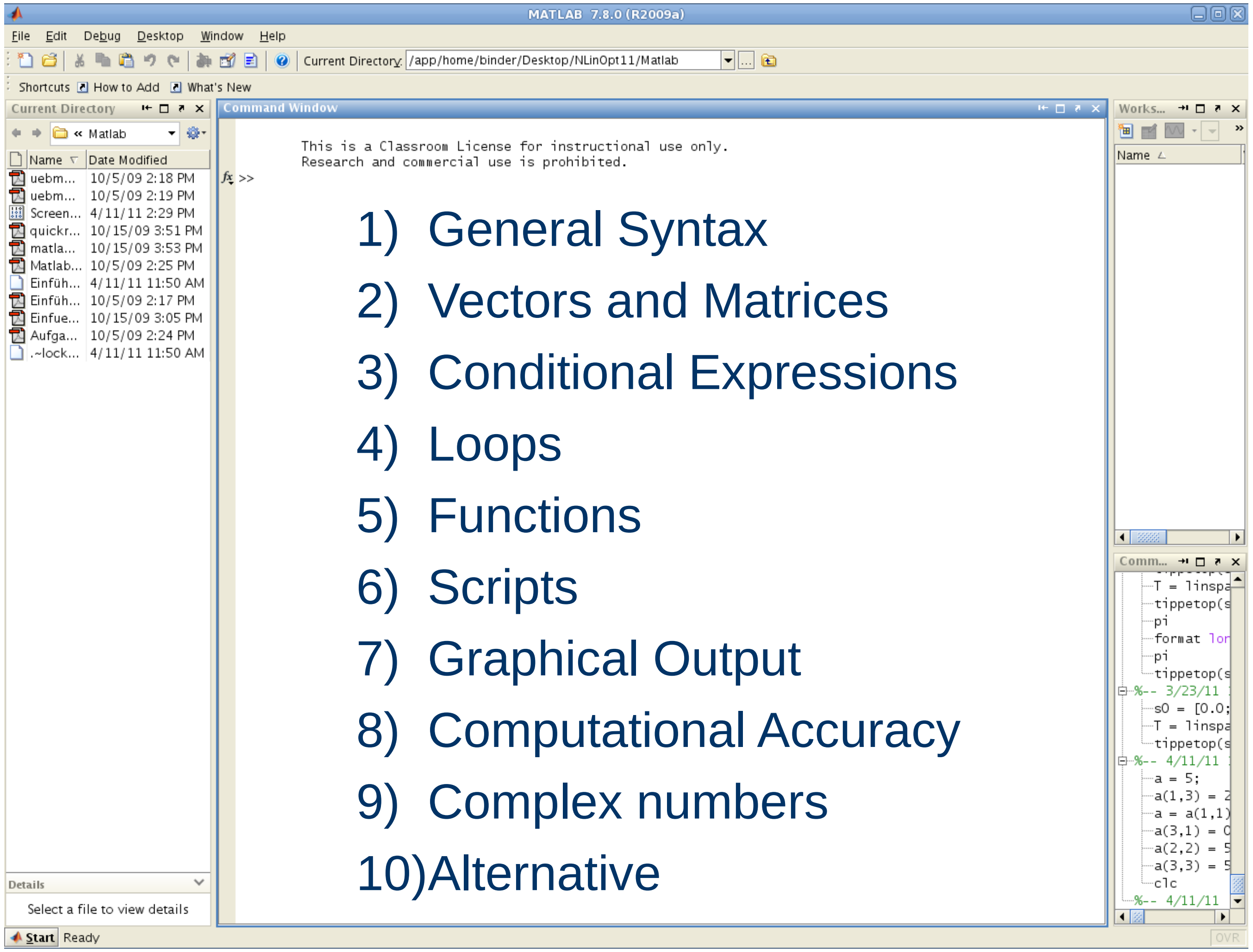
Linear Optimization

WS 2017/2018

Prof. Dr. E. Kostina

M. Sc. H. Stibbe

Contents


MATLAB 7.8.0 (R2009a)
File Edit Debug Desktop Window Help
Current Directory: /app/home/binder/Desktop/NLinOpt11/Matlab
Shortcuts How to Add What's New
Current Directory
Command Window
Works...
Comm...
Details
Select a file to view details
Start Ready
OVR

```
  This is a Classroom License for instructional use only.
  Research and commercial use is prohibited.
fx >>
```

Name	Date Modified
uebm...	10/5/09 2:18 PM
uebm...	10/5/09 2:19 PM
Screen...	4/11/11 2:29 PM
quickr...	10/15/09 3:51 PM
matla...	10/15/09 3:53 PM
Matlab...	10/5/09 2:25 PM
Einfüh...	4/11/11 11:50 AM
Einfüh...	10/5/09 2:17 PM
Einfue...	10/15/09 3:05 PM
Aufga...	10/5/09 2:24 PM
.~lock...	4/11/11 11:50 AM

```
  T = linspa
  tippetop(s
  pi
  format lon
  pi
  tippetop(s
%-- 3/23/11
  s0 = [0.0;
  T = linspa
  tippetop(s
%-- 4/11/11
  a = 5;
  a(1,3) = 2
  a = a(1,1)
  a(3,1) = 0
  a(2,2) = 5
  a(3,3) = 5
  clc
%-- 4/11/11
```

General Syntax

UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

Characteristics of Matlab

- MATLAB = MATrix LABoratory
  ⇒ Matlab determines all variables as matrices

- a = 5;   % (1,1)-matrix
- a(1,3) = 2;   % Expansion of *a* to a row vector *a* = (5,0,2)
- a = a(1,1);   % Recovery of *a* = 5
- a(3,1) = 0;   % Expansion of *a* to a column vector *a* = (5,0,0)^T
- a(2,2) = 5; a(3,3) = 5;
  % Expansion of *a* to a (3,3)-matrix with fives on the diagonal

- The semicolon at the end of the row suppresses the output of the input.

Matlab Help Functions

- If we know, which function we want to use, but we do not remember, what the exact Syntax (in which order we have to fill in the parameters?) of the function is, then we can call up the help in the Command Window:

```
help quadprog
```

- Alternative there is a online-help we can call up on the task bar.
- If we do not know, how the Matlab function we need is called, we can use lookfor. It searches in all the help texts for a keyword and lists all functions that could fit:

```
lookfor 'quadratic programming'
```

Vectors and Matrices

UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

Initialization of Vectors and Matrices

- u = [1 2 3];
  u = [1, 2, 3];   % row vector u = (1 2 3)
- v = [1; 2; 3];   % column vector v = (1 2 3)^T
- M = [1 2 3; 4 5 6];   % matrix with two rows and three columns

- E = eye(n);   % (n,n)-identity matrix
- N = zeros(n,m);   % zero matrix with n rows and m columns
- O = ones(n,m);   % matrix with n rows and m columns filled with 1

- Vectors are matrices, where one dimension equals 1!

The :-Operator --- Simple Application

- For the simple generation of index vectors for example:

u = 1:3	is the same as	u = [1 2 3]
v = (1:3)'	is the same as	v = [1; 2; 3]

- To call up component vectors:

u(1:2)	yields to the first two entries of *u* (row vector)
v(2:3)	yields to the last two entries of *v* (column vector)
M(2, :)	yields to the second row of matrix *M*
M(1:2, 2:3)	yields to a submatrix of *M*, that is made of the entries of the 1. and 2. row of the 2. and 3. column

- Altogether: a:b means “from *a* to *b*”

  : means “all entries of this dimension”

The :-Operator --- double application

- For the additional determination of a different step size besides 1:

w = 0:5:15	is the same as	w = [0, 5, 10, 15]
M(1:5:16, 1:3:16)	yields to the entries of the 1., 6., 11., 16. row of the 1., 4., 7., 10., 13., 16. column of *M*	

- Altogether: a:n:b means “from *a* every *n*-th element (of this dimension) to *b*”

UNIVERSITÄT HEIDELBERG ZUKUNFT SEIT 1386

Calculations

- Transpose a matrix:

  A' yields to *A^T*
- Scalar product of two vectors:

  u'*v yields to the scalar product *u^T v* (equivalent: dot(u,v))
- Cross product (vector product of two vectors):

  cross(u,v)
- Matrix-vector-multiplication:

  A*b yields to the vector *Ab*
- matrix-matrix-multiplication:

  A*B yields to the usual matrix product *AB*
- Inverse of a matrix:

  inv(A) yields to the inverse of *A* (equivalent: A^-1)

UNIVERSITÄT HEIDELBERG ZUKUNFT SEIT 1386

Component-by-component calculations

- A dot in front of the arithmetic operator means, that the operation is done component-by-component:

a^2	yields to the scalar product of *a* with itself
a.^2	yields to a vector with the entries *a(1)^2, ... a(n)^2*
a*b	yields to the scalar product of *a* and *b*
a.*b	yields to a vector with the entries *a(1)*b(1), ..., a(n)*b(n)*

  analogous for matrices.

- Important: the user has to take care himself, that the dimensions of the matrices and vectors fit to each other, otherwise Matlab will generate an error message!

Solving a system of equations

- Assumption: The system of equations Ax=b is uniquely solvable, i.e.
  x = A^-1 * b
- Possibility 1: x = inv(A)*b;

  To calculate a inverse it is very expensive and can possibly need a lot of computation time ⇒ we should avoid this
- Possibility 2: x = A \ b;

  Solves a system of equations with the help of a QR-decomposition (Gauß-algorithm) of the matrix A ⇒ especially if we have big systems, it is much more faster

Conditional Expressions

UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

If ... else ...

- Simple Case query :

```
if (Condition 1)
        Case 1
elseif (Condition 2)
        Case 2
else
        otherwise
end
```

Example:

```
if (k == 1)
        disp('k is 1.');
elseif (k == 2)
        disp('k is 2.');
else
        disp('k is neither 1 nor 2.');
end
```

Switch … case ...

- Alternative case discrimination by numerical values

```
switch Variable
    case Value1
        Case 1
    case Value2
        Case 2
    otherwise
        Otherwise
end
```

Example:

```
switch k
    case 1
        disp('k is 1.')
    case 2
        disp('k is 2.')
    otherwise
        disp('k is neither 1 nor 2.');
end
```

Relation- and logical operators

- relation operator:

Matlab	Bedeutung
<	<
>	>
<=	≤
>=	≥
==	=
~=	≠

- logical operator:

Matlab	Bedeutung
~	¬ (not)
&&	∧ (and)
||	∨ (or)

- The signal words "true" and "false" do not exist in Matlab; it holds: 0 means "false" and every other number means "true"

Loops

Syntax

- There are for- and while-loops

```
for index=start(:stepsize):end
        loop body
end

while Condition
        loop body
end
```

Example:

```
for i=1:2:n
        a(i) = i^2;
end

while i<(n/2)
        a(i) = i^2
        i=i+1;
end
```

- With the command 'break', you can end the loop before the stop criterion is reached

UNIVERSITÄT HEIDELBERG ZUKUNFT SEIT 1386

Alternatives

- Loops often have high computational costs and should be avoided. This is very easy with Matlab

`for i=1:n`
`    A(2,i) = b(i)`
`end`
	is the same as	`A(2,:) = b;`

`for i=1:n`
`    quad(i)=i^2`
`end`
	is the same as	`quad = (1:n).^2`

Functions

UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

Already implemented functions

- In Matlab there are a lot of functions already implemented:
  - Trigonometry: sin, cos, tan, asin, acos, atan, sinh, cosh, tanh
  - System of equations: \, mldivide, ...
  - Optimization: fmincon, quadprog, ...

- A lot of important functions do not exist:
  - Newton method
  - Simplex method

Example (vector.m)

```
% Comment in front of a function call up with a description of the
% function should never be missing, because it can be displayed
% with “help functionsname” and searched with “lookfor ...”
function [vec] = vector(n)
        vec1 = 1:n;                    % Initialization with 1 to n
        vec2 = n+1:2*n;                % Initialization with n+1 to 2n
        vec = [vec1, vec2];            % Assembling
end
```

- The output is the argument in the angular braces (here: vec) and can also directly be assigned to a variable: var = vector(3) yields to var = (1, 2, 3, 4, 5, 6)

inline-functions

- The explicit definition of a new function is not necessary for a simple coherence.
- Simple constructions can be assigned directly:

```
f = inline('x.^3 + 2*x.^2', 'x');
```

  generates the component-by-component function $f(x) = x^3 + 2x^2$
- Rule of thumb: An inline-function has to be one single expression, i.e. no branching or loop or the like.

Scripts

UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

Construction of scripts

- Scripts almost work the same way as functions, but they do not have a name, which can be used to use the script in a different context. That means that every function body without the initial row function[ret] =name(param) and the ending end is also a script.
- Example from above

```
vec1 = 1:n;            % Initialization with 1 to n
vec2=n+1:2*n;          % Initialization with n+1 to 2n
vec =[vec1,vec2];      % Merging
```

is a script for a fixed n; the output is made by e.g. neglecting the last semicolon.

UNIVERSITÄT HEIDELBERG ZUKUNFT SEIT 1386

Scripts vs. Functions

- Scripts are stand-alones and we cannot hand over any parameters. Functions can be invoked from other functions or scripts (possibly with parameters).
- All variables that are generated in a script are saved even after the termination of the script. The variables that are generated in a function have to be returned explicitly and be saved: if we want to use the output of function [a,b]=func(c), we have to explicitly save them: [x,y]=fun(z).
- Scripts also collect variables that we do not need anymore. Those just overload the storage. The command clear can help in this case.

Graphical Output

UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

Plots

- Function graphs can be displayed with the plot-comand:

```
x = -5:0.5:5;
f = inline('exp(-abs(x))', 'x');
plot(x, f(x), '-r')
```

- The last parameter of the plot-function determines the look of the output: '-r' displays the graph with a drawn through red line.

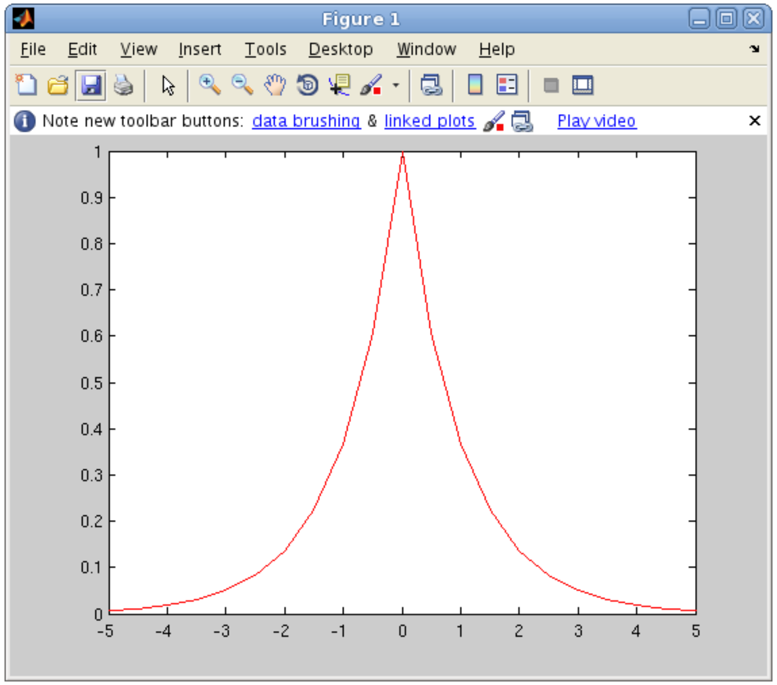


Figure 1
File Edit View Insert Tools Desktop Window Help
Note new toolbar buttons: data brushing & linked plots
Play video
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-5
-4
-3
-2
-1
0
1
2
3
4
5

Graphic Export and Pixel Graphic

- Graphics that are made with plot can be exported in several graphics formats. This can either be done with a toolbox that is available in the graphic windows or with the command saveas.
- Matlab can also interpret every graphic as a matrix with every pixel being a matrix entry and every number value a color (depending on the use colormap). Conversly, every arbitrary matrix can represent a graphic.
- The corresponding commands are imread (graphic as matrix) and imagesc (matrix as graphic). Sample graphics to demonstrate different colormaps can be found using the command imageext.

Computational Accuracy

Computational and Output Accuracy

- By default the output of a calculation is shown with 4 decimal places.
- The internal computational accuracy is greater, around 16 decimal places.
- The output and presentation of results can be changed using the command format:

format short	format long
pi = 3.1416	pi = 3.141592653589793
format short e	format long e
pi = 3.1416e+00	pi=3.141592653589793e+00
(4 decimal places	15 decimal places)

Complex Numbers

UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

Syntax

- Matlab automatically calculates with complex numbers if a result is not real anymore:

  sqrt(-4) works and yields to the result 0+2.0000i
- The input of complex numbers can be made in different ways:

  i. z = 1.234 + 5.678i;

  ii. z = complex(1.234, 5.678);

  iii. a = 1.234; b=5.678;

  ```
  z = a+b*1i;        % it has to be '1i' and not just 'i' to avoid
                       confusion with running indices in loops
  ```

UNIVERSITÄT HEIDELBERG ZUKUNFT SEIT 1386

Complex Conjugation and Polar Graph

- The complex conjugated number can be obtained with the command conj:
  z=1.234 + 5.678i; conj(z) yields to 1.2340-5.6780i
- For the calculation of the scalar product of two complex numbers with dot(z1,z2) Matlab automatically uses the complex conjugated of the first number.
- A polar display of complex numbers can be displayed with the command compass.

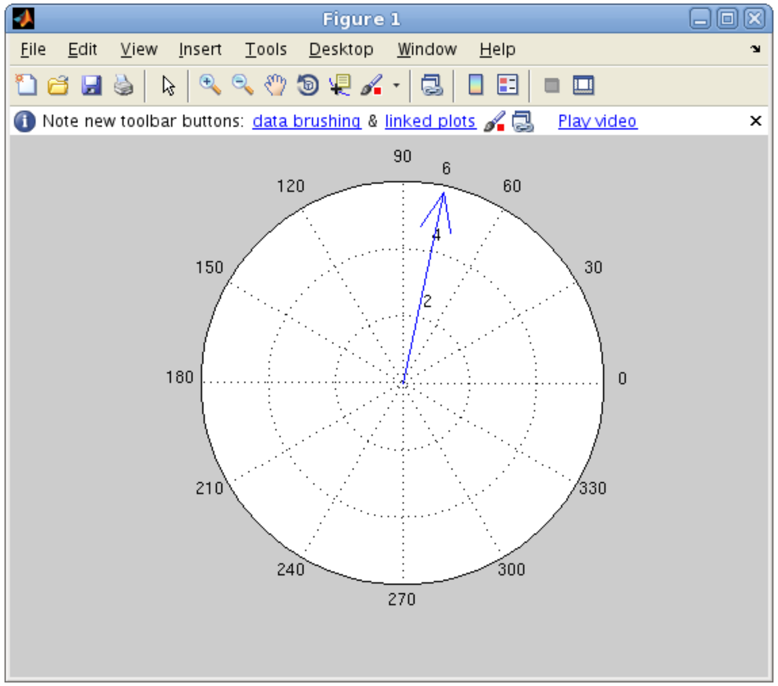


Figure 1
File Edit View Insert Tools Desktop Window Help
Note new toolbar buttons: data brushing & linked plots Play video
90
60
30
0
330
300
270
240
210
180
150
120
2
4
6

UNIVERSITÄT HEIDELBERG
ZUKUNFT SEIT 1386

Alternative

UNIVERSITÄT HEIDELBERG
ZUKUNFT SEIT 1386

Alternative for Matlab

- Matlab is a fee-based program, that is available on

  http://www.mathworks.com

  also in a more cost-effective student version.
- Free student licenses are offered by the Software-Portal of Asknet

  https://urz.asknet.de/cgi-bin/home
- A complementary alternative to Matlab is Octave

  (http://www.gnu.org/software/octave/).
- This is widely compatible to Matlab, but not totally; i.e. Software that is written in Octave often runs in Matlab unchanged, but not always. Not for all operating systems there is a suitable graphical user interface.