Friedrich-Alexander-Universität Erlangen-Nürnberg

Lab Course

Basics on MATLAB

International Audio Laboratories Erlangen

Prof. Dr. Meinard Müller
Friedrich-Alexander Universität Erlangen-Nürnberg
International Audio Laboratories Erlangen
Lehrstuhl Semantic Audio Processing
Am Wolfsmantel 33, 91058 Erlangen
meinard.mueller@audiolabs-erlangen.de

AUDIO LABS

International Audio Laboratories Erlangen
A Joint Institution of the
Friedrich-Alexander Universität Erlangen-Nürnberg (FAU) and
the Fraunhofer-Institut für Integrierte Schaltungen IIS

Fraunhofer IIS

Authors:

Meinard Müller
Stefan Balke
Jonathan Driedger
Thomas Prätzlich

Tutors:

Meinard Müller
Stefan Balke
Jonathan Driedger
Thomas Prätzlich

Contact:

Meinard Müller
Stefan Balke
Jonathan Driedger
Thomas Prätzlich
Friedrich-Alexander Universität Erlangen-Nürnberg
International Audio Laboratories Erlangen
Lehrstuhl Semantic Audio Processing
Am Wolfsmantel 33, 91058 Erlangen
meinard.mueller@audiolabs-erlangen.de
stefan.balke@audiolabs-erlangen.de
jonathan.driedger@audiolabs-erlangen.de
thomas.praetzlich@audiolabs-erlangen.de

Lab Course

Basics on MATLAB

Abstract

This document gives a small introduction to MATLAB. Rather than being comprehensive, we only introduces some basic functions that are needed in the subsequent lab courses.

1 General

Call the help menu by `help` to get an overview of the several topics. By typing `help topic` you get more information about `topic`. Some examples:

`elfun`	Elementary math functions.
`elmat`	Elementary matrices and matrix manipulation.
`matfun`	Matrix functions - numerical linear algebra.
`ops`	Operators.
`paren`	Parentheses, braces, and brackets.

2 Basic Commands to Get Around

One important source of information is the homepage `www.mathworks.com` of the developer MATHWORKS, where you can find a variety of tutorials for different purposes. Furthermore, you should try out the following important general MATLAB functions:

`ans`	Most recent answer.
`clc`	Clears the command window and homes the cursor.
`cd`	Change current working directory.
`clear`	Clear variables and functions from memory.
`dir`	List directory.
`help`	On-line help, display text at command line. `help`, by itself, lists all primary help topics.
`help` topic	Information concerning the topic (e. g., "`help elmat`").
`lookfor` keyword	Keyword search all M-files for keyword (e. g., "`lookfor Fourier`").
`open` file.m	Opens m-file (e. g., "`open fftfilt.m`").
`addpath`	Adds a new directory to the working path.
`quit`	Quit MATLAB session.

3 Vectors and Matrices

Type in the following commands and understand their mechanics.

```
>> a = 0:0.1:0.55
a =    0    0.1000    0.2000    0.3000    0.4000    0.5000

>> a = 1:10
a =     1     2     3     4     5     6     7     8     9    10
```

```
>> b = [1 zeros(1,2)]
b =    1     0     0

>> c = [a(3:5); b; ones(1,3)]
c =  3     4     5
     1     0     0
     1     1     1

>> c'
ans = 3     1     1
      4     0     1
      5     0     1

>> c*c
ans = 18    17    20
       3     4     5
       5     5     6

>> c.*c
ans = 9    16    25
      1     0     0
      1     1     1

>> c(2,:)=[ ]
c =  3     4     5
     1     1     1

>> d = repmat(c,2,3)
d =  3     4     5     3     4     5     3     4     5
     1     1     1     1     1     1     1     1     1
     3     4     5     3     4     5     3     4     5
     1     1     1     1     1     1     1     1     1
```

4 Processing Vectors and Matrices

The following commands are very useful to sort or to threshold vectors or matrices. Use the `help` function for further information on the commands.

```
>> x = rand(1,8)
x = 0.3311  0.5629  0.4206  0.4007  0.0548  0.0213  0.4543  0.0578

>> y=sort(x)
y = 0.0213  0.0548  0.0578  0.3311  0.4007  0.4206  0.4543  0.5629

>> abs(x)<0.4
ans =   1     0     0     0     1     1     0     1

>> i = find(abs(x)<0.4)
i =  1     5     6     8

>> x(i) = zeros(size(i))
x = 0       0.5629   0.4206  0.4007  0       0       0.4543  0
```

5 Basic Functionalities

Familiarize yourself with the following functionalities of MATLAB.

- Arrow keys ↑ und ↓: Scroll through command history.
- Commandprefix in the command line for restricted history browsing.
- MATLAB as calculator: `3*2/3-5^2`
- Variables (take care of upper- and lower-case: `x=10, X='test'`
- Types: `int, real, complex, inf (1/0), NaN`
- Functionality of comma (,) and semicolon (;)
- Functions (`help elfun`): `sin, cos, sin, tan, exp, acos, atan, log, log10, sinc, sqrt, ...`
- Row vectors: `v=[1 2 3], w=[3:0.5:4 0:0.1:0.45]` (→ `help colon`)
- Column vectors: `v = [1; 2; 3]`
- Transposition: `v=w'`
- Length of a vector: `length(v)`
- Spaces in MATLAB: `v1=[0+ 1 2 3], v2=[4 +5 6], v3=[4+5 6]`
- Chaining: `w1=[v1 v2], w2=[v1;v2], w3=[v1' v2']`
- Sorting: `sort(v)`
- Accessing vector elements: `v=(1:10); v(2), v(2:2:5)`
- Addition: `v1+v2`
- Scalar multiplication: `3*v1`
- Inner product: `v1*v2'` oder `dot(v1,v2)`
- Pointwise operations: `v1.*v2, v1./v2, v1.^2, ...`
- Norm: `norm(v1)` oder `sqrt(v1*v1')`
- Convolution: `conv(v1,v2)`
- Complex numbers: `x=[1+3i, 2-2i]`
- Complex conjugate and transposition: Compare `x'` and `x.'`
- Functions on complex data: `real, imag, isreal, conj, abs, angle`
- Matrices: A=[1 2 3 4; 5 6 7 8; 1 2 3 4; 8 7 6 5]
- Dimensions of a matrix: `size(A), size(A,1), size(A,2)`
- Matrix functions: `eye, zeros, ones, diag, rand, inv, convn, ...`
- Accessing matrix elements: `A(3,3)=A(15)+3*A(2,2), A(:,2:3)`
- Audio reading and writing functions: `wavread, wavwrite, ...`

- Loops: `for, while`
- Conditionals: `a=rand(1);b=rand(1);if a<b b, elseif a==b '=', else a, end`
- Further MATLAB-functions: `round, fix, floor, ceil, sign, rem, mod, sum, max, min, find(y>0)`, ...
- M-files
- `F5` Runs the currently opened script.
- `F9` Executes the marked commands in a script.
- `Ctrl + C` Stops the current computation.
- `Ctrl + Enter` Executes subpart of Script encapsulated by `%%..%%`.
  Example:
  ```
  %% Part 1
  % Code for Part 1
  ...
  %% Part 2
  ...
  ```
- `Ctrl + D` Opens the M-file of the currently marked function call in a script.

6 Lab-related functions

Familiarize yourself with the following functionalities of MATLAB.

spectrogram

Type `help spectrogram`

Example:

```
% generate a sin wave
fs = 22050;             % sampling rate
d = 2;                  % duration in seconds
t = (0:d*fs-1)/fs;      % time in seconds
A = 0.5;                % amplitude
f = 400;                % frequency
x = A * sin(2*pi*f*t);  % function

% compute the spectrogram
win = hann(1024);
noverlap = 512;
X = spectrogram(x,win,noverlap);
```

median

Type `help median`

Example:

```
b = [ 1 7 4 3 193 9 1];
median(b)

B = [ 4 3 193 9 1; 5 1 200 2 3; 398 401 389 420 411; 4 2 198 8 5];
median(B,1)
median(B,2)
```

7 Visualization

Familiarize yourself with the following code examples for visualization.

plot

The `plot` command can be used to visualize one dimensional data such as a waveform.

```
% generate a sin wave
fs = 22050;              % sampling rate
d = 2;                   % duration in seconds
t = (1:d*fs)/fs;         % time in seconds
A = 0.5;                 % amplitude
f = 440;                   % frequency
x = A * sin(2*pi*f*t);   % function

figure;
plot(x);                 % plot without a specified time-axis

figure;
plot(t,x);               % plot on the specified time-axis
xlabel('time in seconds');
ylabel('amplitude');
xlim([0.5 1.5]);
ylim([-1 1]);
```

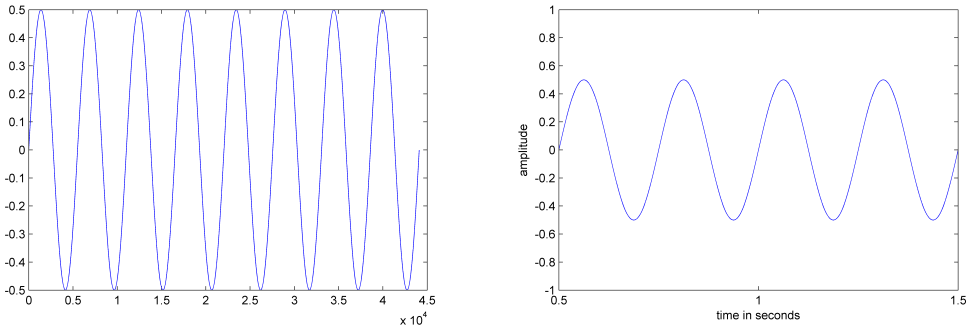

Figure 1: Generated Figures.

imagesc

The `imagesc` command can be used to visualize two dimensional data such as a spectrogram.

```
% generate a sin wave
fs = 22050;             % sampling rate
d = 2;                  % duration in seconds
t = (0:d*fs-1)/fs;      % time in seconds
A = 0.5;                % amplitude
f = 4;                  % frequency
x = A * sin(2*pi*f*t);  % function

% compute a spectrogram
win = hann(1024);
noverlap = 512;
X = spectrogram(x,win,noverlap);

figure;
imagesc(abs(X));

tX = (0:size(X,2)-1)/(fs/(1024-512)); % time axis of the spectrogram
fX = (0:size(X,1)-1)*(fs/1024);       % frequency axis of the spectrogram
figure;
imagesc(tX,fX,abs(X));
xlabel('time in seconds');          % add a label for the x axis
ylabel('frequency in Hertz');       % add a label for the y axis
colorbar;                           % add a colorbar
colormap(1-gray);                   % use an inverted colormap
axis xy                             % flip the direction of the y axis
ylim([0 1500]);                     % show just the frequency range from 0 to 1500 Hz
```

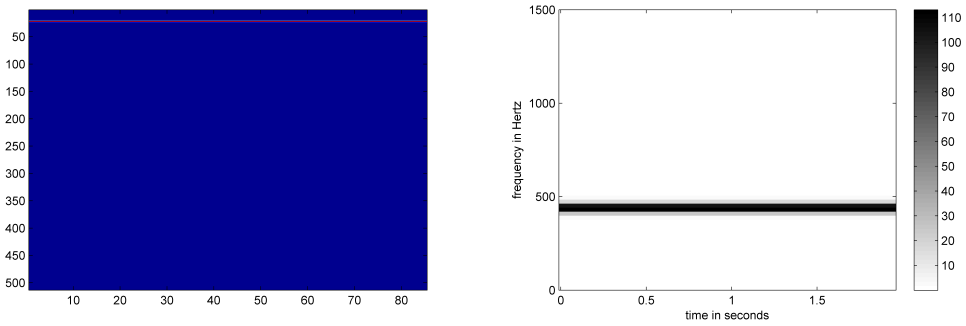

Figure 2: Generated Figures.