Laboratory Test

DSP 2020-2021

Information

- The test is taken in the lab, and lasts for 1 hour
- General Matlab stuff you need to know is listed in the **Syllabus** section. This is not an exhaustive list. There may be things I forgot to put in the list, but the basics are there.
- Template subjects (i.e. exercises extracted from the labs) are in **Template Subjects** section
- The test will be roughly based on these templates, with modifications

Syllabus (not exhaustive)

- Define scalars, vectors, matrices
  - Generate constant vectors (zeros, ones, some value)
  - Generate sin and cos signals of a certain length, with various amplitude, frequencies and initial phase
  - Generate random numbers, vectors and matrices, using `rand` and `randn`
  - Generate linearly spaces values between a start value and a stop value (e.g. `linspace()` or `start:step:stop`)
- Concatenate vectors and matrices. Build longer signals from more parts.
- Access elements from vectors/matrices. Select rows or columns from matrices.
- Do basic mathematical operations with scalars, vectors, matrices
- Display a text message (`disp()` or `fprintf()`)
- Use simple instructions (`if`, `for`, `while`)

- Make plots
  - Plot a vector
  - Plot a vector against another vector (e.g. put the vector `n` on the horizontal axis)
  - Plot multiple signals on the same figure
  - Use subplots
- Operate with audio files (`.wav`, `.mp3`)
  - load a file
  - load only a certain part of a file (e.g. the first 5 seconds)
  - extract a single channel
  - play with the data, e.g. swap channels
- Operate with images
  - Create a simple grayscale or color image and display it
  - Load and display an image file. Also do simple adjustments (convert to grayscale, divide to 255).
  - Apply a simple filter to an image (e.g. a $3 times 3$ matrix with coefficients)
- Create and display a video based on a simple animation of an image (e.g. scrolling the image, change luminosity)
- Create and use Matlab functions
- Implement a system in a Matlab function, based on the system equation
  - the function takes `x` as input and outputs the result vector `y`
- Pass a function as argument to another function and use it inside
- Do convolutions of vectors with `conv()`
- Compute the Discrete Fourier transform (DFT) of a vector, compute and display the modulus and the phase of the Fourier transform
- Implement a system in Simulink based on the equation
  - Also apply some input, visualize the output
  - Apply an impulse and visualize the impulse response
- Test linearity ~~and time-invariance~~ of systems in Simulink
- Design a low-pass/high-pass etc. filter with `fdatool` GUI, and implement it in Simulink (Lab 11)
- Design an oscillator with a prescribed frequency, and implement it in Simulink (Lab 11)

Template Subjects

Lab 2

1. Plot on the same figure the signals $sin(2\pi ft + \frac{\pi}{4})$ and $cos(2\pi ft + \frac{\pi}{8})$, with $f = 0.3$ and $t \in [0, 10]$.
2. Load the audio file ‘Kalimba.mp3’ in the Matlab workspace. Only load samples between 1 and 200000 (to avoid out of memory error)
   a. Play it through the computer’s audio device
   b. Change the sampling frequency to half the correct value, and play again. How will the sound be changed?
   c. Amplify the sound by multiplying the data by 4. Play the sound and observe the difference.
   d. Swap the left and right channels (it’s a stereo file) and play the sound again.

Lab 3 version 1

3. Create a color image representing the Romanian flag (3 stripes of blue, yellow, red). Create the image using the following steps: Create three matrices for the R, G, B components of the image Concatenate the three matrices across third dimension, into a 3D tensor

Variant: Make another simple flag or figure, color or grayscale, and show it

Lab 3 version 2

1. Load the `Lena` image (use `imread()`), convert it to double, convert it to grayscale, scale the values to the [0, 1] range, and display the image (use `imshow()`).
2. Construct a new image based on the `Lena`, but in which each pixel value is set as a linear combination of the original pixels around it, as in the following equation:

$$\begin{aligned} y[i,j] = \frac{1}{9}x[i-1,j-1] + \frac{1}{9}x[i-1,j] + \frac{1}{9}x[i-1,j+1] \\ +\frac{1}{9}x[i,j-1] + \frac{1}{9}x[i,j] + \frac{1}{9}x[i,j+1] \\ +\frac{1}{9}x[i+1,j-1] + \frac{1}{9}x[i+1,j] + \frac{1}{9}x[i+1,j+1] \end{aligned}$$

   Ignore the first and last row/column, if needed.

   Display the resulting image in a new window. How did it change?

Lab 4

1. Load the `Lena` image (use `imread()`), convert it to a grayscale image, convert it to `double` type, adapt the values to the $[0, 1]$ range, and display it (use `imshow()`).
2. Create a video sequence by scrolling the Lena image circularly to the right, by 3 pixels at every frame. Display the video at 25fps.

Code template for creating a video sequence in Matlab:

```
height = ...; % desired height
width  = ...; % desired width
NoF    = ...; % desired number of frames
% an array of size height x width x 1 x NoF:
video  = zeros(height, width, 1, NoF);
for i = 1:NoF
    video(:,:,:,i) = ... the frame number i ... ;
end

% Play the sequence
implay(video);
```

Variant: do another thing instead of scrolling, like change luminosity etc.

Lab 5

1. Create a function `mysys1()` that implements the following system H_1:

$$y[n] = H_1\{x[n]\} = \frac{1}{4}x[n] - \frac{1}{2}x[n-1] + \frac{1}{4}x[n-2]$$

   - the function takes 1 input argument `x` and outputs 1 result vector `y`

2. In a separate script, test the linearity of this system in the following way:
   - generate two random vectors `x` and `y` and two random numbers `a` and `b`
   - apply the function `mysys1()` to `a*x`, `b*y`, and `a*x + b*y`, and check if the results verify the linearity equation
   - display a message indicating if the system is linear or not linear

Lab 6

1. Create a Simulink model to implement the following system H_1:

$$y[n] = H_1\{x[n]\} = \frac{1}{4}(x[n] + x[n-1] + x[n-2] + x[n-3])$$

   - the system should be implemented as a Subsystem block with one input and one output signal

2. Visualize the impulse response of the system

   - add a unit impulse as the input (hint: can be created from two unit ramp blocks, delayed)
   - add a Scope at the output to visualize the data

Lab 7 Variant 1

1. Create a Simulink model to implement the following system H_1:

$$y[n] = H_1\{x[n]\} = 0.8y[n-1] + 0.25x[n] + 0.1x[n-1]$$

   - the system should be implemented as a Subsystem block with one input and one output signal

2. Test linearity of this system as follows:

   - create three copies of the system inside the model (copy/paste)
   - use two random input vectors `x` and `y` (use two *Random* blocks)
   - apply input signals `x`, `y` and `x+y` to the three copies of the system
   - add the outputs of the systems which have `x` and `y` as inputs, then subtract the output of the system which has `x + y` as input
   - show the resulting signal. Is the system linear?

~~Lab 7 Variant 2~~

~~Same thing, but test time invariance instead of linearity (Ex. 5):~~

- ~~the system will be applied to an input vector `x`, and to `x` prepended with a variable number of zeros (i.e. time delayed)~~
- ~~the outputs shall be checked if they verify the time invariance equation~~

Lab 8

1. Load an audio signal and extract a 10 seconds long sequence of it.

   a. Convolve the sequence with the impulse response $\{1/6, 1/6, 1/6, 1/6, 1/6, 1/6\}$. Play the resulting sequence.

   b. Load another impulse response from the file "Scala Milan Opera Hall.wav" (use `audioread()`). Call the resulting vector **`h`**. Convolve the original audio signal with **`h`** and play the result.

   c. Convolve the result from b) with another impulse response from the pack. Play the resulting signal.

   d. Compute and display the equivalent impulse response of the complete system in points b) and c).

Lab 9

1. Generate a 100 samples long signal `x` defined as $x[n] = 0.7\cos(2\pi f_1 n) + 1.2\sin(2\pi f_2 n)$, with $f_1 = 0.05$ and $f_1 = 0.1$.

   a. Plot the signal in the top third of a figure (use `subplot()`).

   b. Compute the Fourier series coefficients with `fft()` and plot their magnitude in the middle third, and their phase in the lower third.

   c. Repeat the plot but do the FFT in N=1000 points (use `fft(x, N)`. What changes?

Lab 11 Variant 1

1. Use the Filter Design tool in Matlab (`fdatool`) to design a IIR high-pass filter with order 3, with cutoff frequency 0.07. Implement the filter in Simulink and then apply at the input the signal $x[n] = \cos(2\pi 0.03n) + \cos(2\pi 0.18n)$ and visualize the output $y[n]$. Compare with the input signal.

Lab 11 Variant 2

2. Use the Filter Design tool in Matlab (`fdatool`) to design an oscillator with frequency 0.05. Implement it in Simulink, visualize & play the output signal.

   Use the following steps to design the oscillator:

   1. design a system of order 2 with 2 conjugate poles placed **on the unit circle** at the correct frequency, and 2 zeros at low & high frequencies
   2. implement the system in Simulink, **omitting the input signal** (not necessary)
   3. set a non-zero initial condition in the system, to start-up the oscillator