Laboratory Test

DEDP 2020-2021

Information

- The test will last for 1 hour
- You will upload the Matlab files on Moodle (or send by email)
- General Matlab stuff you need to know is listed in the **Syllabus** section
- 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

Things to know in Matlab:

- Load and save a `*.mat` file
- Generate random numbers with Gaussian and uniform distributions
  - single numbers, vectors or matrices
  - with various distribution parameters (mu, sigma, a, b)
- Generate a random vector with 0's and 1's in variable proportions (or some value A instead of 1)
- Generate a sin or cos signal of a certain length, with a specified amplitude, frequency and initial phase
- Generate a vector with N values equally spaced between a start and a stop value (e.g. `linspace()`)
- Compute mean and variance of vectors or columns

- Operate with columns (or rows) of a matrix:
  - extract one or more columns
  - arithmetic operations: add columns, divide element by element, etc
  - same with rows instead of columns
- Count how many values of 1 are in a vector (or maybe how many values equal to A)
- Count pair of values in two vectors (e.g. when there is a 0 in a vector and 1 in another vector, like for false alarms, misses etc)
- Compute Euclidean distance between vectors
- Sort a vector and find the original positions of the smallest k values
- Find the maximum value in a vector and its position
- Find the minimum value in a vector and its position
- Plot a vector
- Plot a vector as a function of another vector
- Create a histogram plot
- Create and use a Matlab function
- Display a message with `fprintf()`
- Create and use simple cell arrays
- General instructions: if, for, etc

Template Subjects

Lab 1

1. Create a Matlab function `myPDF()` that estimates the probability density function from a vector of data.

- the function requires three arguments and returns one value: `p = myPDF(v,x,epsilon)`
- `v` is a vector, and `x`, `epsilon` are scalar numbers
- the function computes how many elements from `v` are in the interval $[x - \epsilon, x + \epsilon]$, divided to the total number of elements of `v`, and also divided to 2 times `epsilon`

2. Plot the probability density function estimated from a vector of data

- generate a vector v with 100000 values from the normal distribution $\mathcal{N}(2, 2)$ and plot the values
- generate a vector n of 50 values uniformly spread between -5 to 15
- apply `myPDF()` on v to estimate the probability density at every value from n (use `epsilon = 0.1`)
- plot the results of the function against the values of n

Lab 2

1. Load the file `ElectionsData.mat`. It contains election data for the local elections in the city of Iasi held on 27.09.2020 (data taken from https://prezenta.roaep.ro). The file contains two variables:

- names: a cell array with the names of the voting centers
- values: a matrix with the voting numbers for each center

The structure of the values matrix is as follows:

- first column: total number of registered voters on permanent lists
- second column: total number of registered voters on complementary lists
- third column: number of votes from permanent lists
- fourth column: number of votes from complementary lists
- fifth column: number of votes from supplementary lists
- sixth column: number of votes with mobile urns

a. Compute the **turnout** for every voting center, defined as: total number of votes / total number of registered voters on all lists.
b. Plot the turnout vector
c. Compute the mean and the variance for the turnout across the city of Iasi.

Lab 3

1. Simulate threshold-based detection with a single sample, as follows:

   - Generate a vector of 100000 values 0 or A, with equal probability (consider $A = 5$)
   - Add over it a random noise with normal distribution $\mathcal{N}(0, \sigma^2 = 1)$
   - Pick a value of $T = A/2$ and compare each element with T to decide which sample is logical 0 or logical 1 (A)
   - Compare the decision result with the true original vector, and count how many correct detections and how may false alarms have been
   - Estimate the four probabilities by dividing the above numbers to the size of the vector

Variant: Same exercise, but written as a function which accepts T as input and return the four values as outputs. Running the function for 100 values of T uniformly spaced between 0 and A, and plotting the resulting vector `phit` against `pfa`.

Lab 4

1. Implement a function `[class] = myKNN(signal, k, trainset)` for performing k-NN classification of a signal:
   - the function takes as input an unclassified signal `signal`, the parameter value `k`, and the training set matrix `trainset`
   - the function computes the Euclidean distance between `signal` and each vector from the training set
   - the output `class` is defined by the majority of the k nearest neighbours of the signal
2. Call the function `myKNN` for the first signal from the testing set and determine its class. Use different values for k: $k = 1$, then $k = 5$, then $k = 15$.

Note: the training set matrix can be loaded from the file `ECG_train.mat`, and the test set from `ECG_test.mat`

Lab 6

1. Generate a 500-samples long sinusoidal signal $s_\Theta = A * \sin(2\pi f n)$ with frequency $f = 0.02$, and add over it normal noise with distribution $\mathcal{N}(0, \sigma^2 = 0.5)$. Name the resulting vector `r`. Plot the `r` vector.
2. Estimate the frequency $\hat{f}$ of the signal via Maximum Likelihood estimation from the `r` vector:
   - Generate 1000 candidate frequencies f_k equally spaced from 0 to 0.5
   - Compute the Euclidean distance between `r` and the sine signal with each candidate frequency
   - Maximum Likelihood: choose $\hat{f}_{ML}$ as the candidate frequency which minimizes the Euclidean distance
   - Display the estimate value $\hat{f}_M L$
   - Plot a sinusoidal with the estimated frequency $\hat{f}_{ML}$, and the original vector `r`, on the same figure

Variant: estimate amplitude A instead of frequency f, in the same way