Laboratory Test

DEPI

Information

- There are 6 subjects in all, shown below according to the laboratory they were done in.
- The test will last for 50 minutes (~10 students in 1st hour, ~10 students in the second hour)
- Your schedule will be available in `ProgramareTest.pdf`.

Subjects

Lab 1

1. Create a Matlab function `myPDF()` that directly 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, `x`, `epsilon` and `p` 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*`epsilon`
2. Plot the probability density function estimated from a vector of data
   - generate a vector v with 10000 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 = 0.3`)
   - plot the results of the function against the values of n

Lab 2

1. Simulate threshold-based detection with a single sample, as follows:
   - Generate a vector of 100000 values 0 or $A = 5$, with equal probability (hint: use `rand()` and compare to 0.5);
   - Add over it a random noise with normal distribution $\mathcal{N}(0, \sigma^2 = 1)$;
   - Compare each element with $T = A/2$ 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 rejections, falsa alarms, misses and correct detections have happened;
   - Estimate probability of correct rejection, false alarm, miss and correct detection (by dividing the above counters to the size of the vector).

Lab 3

1. Simulate the BPSK sender and channel
   - Generate a vector **`x`** of 1000 values 0 or 1, with equal probability (hint: use `rand()` and compare to 0.5).
   - Generate a vector **`s`** of 100000 values as follows:
     - for each bit 0 in **`x`**, put a 100-long sine $A \sin(2\pi f n)$ in **`s`**
     - for each bit 1 in **`x`**, put a 100-long inverted sine $-A \sin(2\pi f n)$ in **`s`**
     - Use $A = 1$, $f = 1/100$.
   - Plot `s`.
   - Generate a vector of white gaussian noise with distribution $\mathcal{N}(0, \sigma^2)$, the same length as `s`, and $\sigma^2 = A/10$.
   - Add the noise to the signal, store result as **`r`**.
   - Plot the resulting signal `r` (do not overwrite the previous figure of `s`, we want to see both)

Lab 4

1. Implement a function `[class] = myKNN(image, k)` for performing k-NN classification of an image:
   - the function takes as input an image `image`
   - the function loads the training set from `trainset.mat` (will be provided)
   - the function computes the Euclidean distance between `image` and each image from the training set
   - the output `class` is defined by the majority of the k nearest neighbours of the image
2. Load the test image matrix `I` from 'testimage.mat' and call the function `myKNN` to determine its class. Use different values for k: $k = 1$, then $k = 5$, then $k = 15$.

Lab 5

1. Load the color image 'peppers.jpg' using `imread()`. Convert the image to `double` and display it (don't convert to grayscale, leave the colors).
2. Use Matlab's k-Means algorithm to cluster all the pixel values (each pixel = a group of three values R, G, B) into 4 groups.
3. Replace each pixel of the image with the *centroid* of its class. Display the image.

Lab 6

1. Generate a 500-samples long sinusoidal signal `x` with frequency $f_0 = 0.01$, 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 $\hat{f}_M L$