Pattern and Speech Recognition Tutorial
Exercise 6

In this exercise you will implement the EM algorithm to find the parameters of a Gaussian Mixture model.

Exercise 1 (10 points)

We can summarize GMM as a three step procedure. An initialization step and a series of two steps, the Expectation step and the Maximization step. You will create a GMM function which takes 2 inputs, the data and the number of the clusters. Inside that function you will perform the three steps. The output must be the final means, the final covariance matrices and the final priors.

1. Load the data from the previous exercise: `https://www.dropbox.com/s/bg2f4u1ujd9qpkj/corpus?dl=0` . For the users of Matlab you should use *fread* and the data type is **float**; for the users of python/numpy, you should use *numpy.fromfile* and the data type is **numpy.float32**.

2. Keep only every 100th sample. Perform PCA, keep the first two dimensions of your transformed data and plot it.

3. Implement a function [means, covarianceMatrices, priors]=GMM(data,numberOfClusters). You can choose a different output form, e.g. a struct.

4. The first step in this function should be the initialization step. For each Gaussian choose 5 random points from the data and set the initial mean to be the mean of those points. Set the mixture components to be $\frac{1}{k}$. Initialize the covariance matrices to identity matrices. Each of the k covariance matrices is a 2×2 matrix. The means are 2×1 column vectors. The priors are scalars.

5. In the Expectation step, you must compute the weighted contribution of every point in every cluster (denoted as gamma in the lecture slides). p_k is the prior of cluster k, P() is the probability of a point in cluster k. Use mvnpdf() from matlab or from *scipy.stats* import multivariate_ normal to compute P(). You can also compute it by your own implementation.

6. In the Maximization step, update your means, covariances and priors. The update rules for the mean and covariance are on page 39 of the lecture slides. To update the prior for each cluster, add the weights of all points to that cluster, and divide it by the number of total points. You can assume that the covariance matrices are diagonal (the diagonal contains the variance of each dimension and all other entries are zero). Why can you assume that they are diagonal?

7. Define your own criterion for stopping the EM iterations and use it to end your GMM.

8. Test your implementation with K=2, K=4 and K=10 and multiple runs for every K. Each time plot the clustered data using different colors for every cluster. Present some of your plots. A point is assigned to the cluster with the maximum posterior probability. You have for every cluster its prior and parameters.

9. Describe the clustering.
10. Compare K-means to GMM. What are the main differences with respect to the decision boundaries?

Exercise 2 (3 bonus points)

Some times K-means is used as an initialization for GMM, since EM is sensitive to the initial parameters. Implement GMM2, a function similar to the previous one, but for the initialization step use K-means for a small number of iterations (either yours, or an already implemented one). Use K-means to find an initial clustering, and from this clustering compute the initial parameters for the EM steps. The means are the means of the clusters, the covariances are the covariances of the data in each cluster, and the priors are the number of points in each cluster divided by the total number of points.

Plot again your findings for K=4 and K=10 and compare them to the random initialization.

Submission architecture

You have to generate a **single ZIP file** respecting the following architecture:

```
tutorial1_<matriculation_nb1>_<matriculation_nb2>_<matriculation_nb3>
|
+--- source
|      |
|      +------- file 1
|      +------- file 2
|      +------- ...
+--- rapport.pdf
+--- README.txt
```

where

- **source** contains the source code of your project,
- **rapport.pdf** is the report where you present your solution with **the explanations (!)** and the plots,
- **README** which contains group member informations (name, matriculation numbers and emails) and a **clear** explanation about how to compile and run your source code

The ZIP filename has to be :

`tutorial5_<matriculation_nb1>_<matriculation_nb2>_<matriculation_nb3>.zip`

You have to choose between the following languages **python** or **matlab**. Other languages won't be accepted.

Some hints

We advice you to follow the following guidelines in order to avoid problems :

- Avoid building complex systems. The exercises are simple enough.
- Do not include any executables in your submission, as this will cause the e-mail server to reject it.

Grading

Send your assignment to the tutor who is responsible of your group:

- Gerrit Gromann `gerritgr@gmail.com`
- Sbastien Le Maguer `slemaguer@coli.uni-saarland.de`
- Kata Naszdi `b.naszadi@gmail.com`

The email subject should start with `[PSR TUTORIAL 6]`