## UML computer project 3

- Handed out: 29.04 (Fr)
- Hand in: 15.05 (So) midnight the latest,
  by email to michael.gutmann@helsinki.fi
- You can do the exercises in pairs. Submit in that case only one report. Please write in the report the names and the student numbers of both of you.
- Submit your report as a single pdf containing figures and **discussion**. In the report, explain what you are doing and why you are doing it. Don't put figures in the report without explaining the result they show. The report should be enjoyable to read; remember that the grading will be based on the report.
- Submit the source code as well (separate attachment, not as appendix in the report). The code needs to be such that running it for every exercise will produce the figures in the report.
- Each exercise in the project gives you points. They will, at the end, determine your grade for the computer project. Each exercise in all the assignments gives you an equal amount of points.

## Exercise 1: Gaussian Mixture Model

This exercise is about Section 11.3 in the lecture notes: The Gaussian mixture model.

1. Draw 2000 samples from a two-dimensional mixture of Gaussians, with two clusters as in Figure 11.5. Make a scatter plot, and explain which parameters of the Gaussian mixture model govern the properties of the data.

2. Implement the EM algorithm in Section 11.5, 11.6. The algorithm should work for data of any dimensionality and any number of clusters. It should return the values of the objective function (Eq. 11.21) during the optimization, the estimated covariance matrices $C_c$, the estimated means $\mu_c$, and the estimated cluster probabilities $\pi_c$. Test the algorithm on your data: show the values of the objective function, and compare the estimates with the true values.
   (Note: (1) The algorithm can "blow up" if it puts a Gaussian onto a single data point. If this happens, it's OK to just restart from different initial point. (2) Depending on your implementation, you might take an operation like $log(u)u$. The limit for $u \to 0$ is zero, but (at least in matlab), the numerical result is infinite or "NaN". Using $log(u + 1e^{-10})u$ prevents this from happening.)

3. (After learning of the parameters) For each data point, compute the posterior density $p(r(t)|\mathbf{x}(t))$ in Eq. (11.14). Assign each data point $\mathbf{x}(t)$ to the cluster $\hat{r}(t)$ which gives the largest value of $p(r(t)|\mathbf{x}(t))$ (MAP estimate). Make a scatter plot where the color of each data point indicates the cluster it belongs to according to the MAP estimate (an example is shown in Fig 1).

4. Repeat the above for 4 clusters and a total of only 100 data points: Create artificial data (with randomly chosen means, covariance matrices, are cluster probabilities), and run the EM algorithm from 5 different starting points. If the objectives for the different runs are not all the same, visualize the MAP estimates as above for the run with the highest and lowest objective.

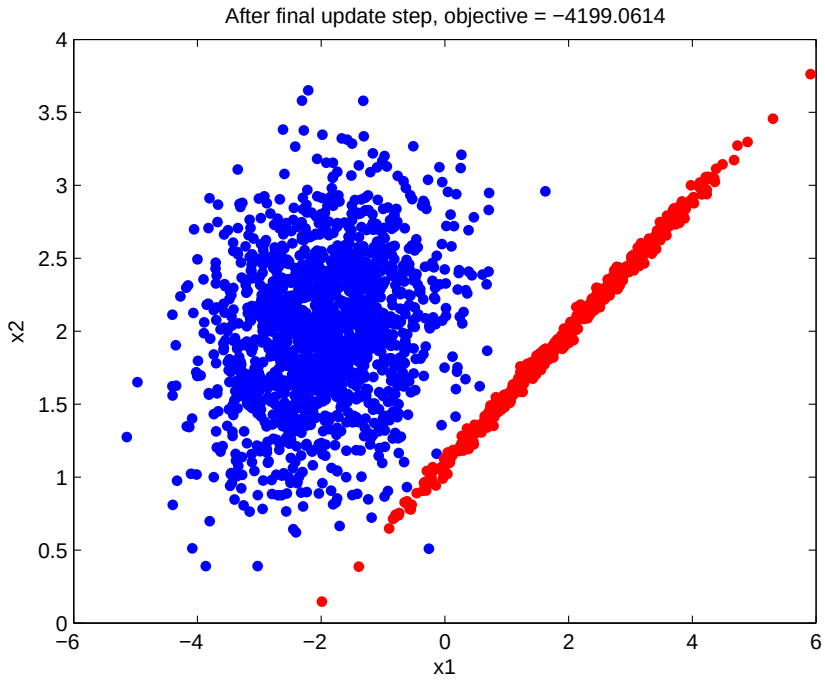


Figure 1: For exercise 1. Example to visualize the MAP estimate of cluster ownership. Red data points are assigned to cluster 1, blue data points are assigned to cluster 2.

## Exercise 2: Nonlinear projection methods

1. Perform linear MDS on the two dimensional data in `data_proj.txt`. Visualize the projection by coloring each data point by the value of its projection. (Make a color plot as Fig.11.2) How does the projection encode the structure of the data?

2. Do PCA on the data: Plot the first principal component direction and compute the first principal components. Color each data point by the value of its first principal component. Compare and relate this kind of projection to the one from the question above.

3. Implement the SOM algorithm on p100 in the lecture notes and apply it to the data. The neighborhood of node $i$ is node $i-1$ and node $i+1$, with period boundaries. This establishes a kind of ordering of the model vectors (cluster means) $\mathbf{w}_i$. Run the algorithm for various numbers of model vectors (about 4 to 20) and various initializations and comment on the result.
   For the visualization, color the data points which are closest to the same model vector with the same color (like for the clustering in the first exercise). Also, plot the location of the model vectors (an example is shown in Fig 2.)

4. Load the images in the folder `images_txt`, and extract from each image nonoverlapping patches of size 10px $\times$ 10px. Note that the images are not all of the same size. Each image will thus provide a different number of patches. Reshape each patch as a 100 dimensional vector, and put all the patches together in one big data matrix of size 100 $\times$ number–of–patches (15696 for the code I wrote). We consider in this exercise each patch as a realization of a 100 dimensional random vector. How do these modeling assumptions differ from those from project2, exercise 3?

5. Apply the SOM algorithm from the question above to the image patches. As preprocessing, remove from each patch the average value and rescale each patch to unit variance (make sure that you avoid division by zero). Try the algorithm with 10, 20, and 30 weight vectors (note, the algorithm may get stuck in local maxima). Visualize the weight vectors as images using the `visual.*` from exercise 1.

6. Set the number of weight vectors to 20 and investigate what changes when you omit in the preprocessing (a) the rescaling of each patch to unit variance and/or (b) the subtraction of the average value from each patch.

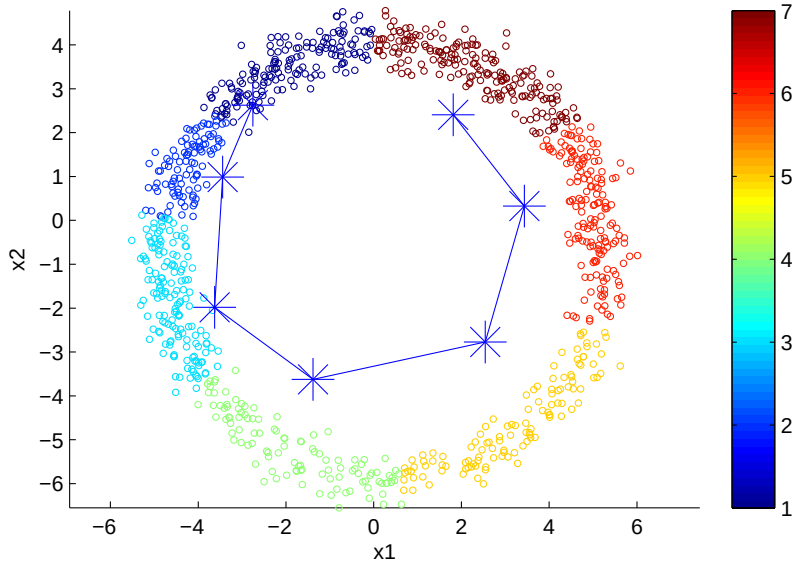


Figure 2: For exercise 2. Example on how to visualize the results of a SOM with seven model vectors.