Gaussian Mixure Models and Expectation Maximization

1 Introduction

The goal of the assignment is to use the Expectation Maximization (EM) algorithm to estimate the parameters of a two-component Guassian Mixture in two dimensions. This involves estimating the mean vector μ_k and covariance matrix Σ_k for both distributions as well as the mixing coefficients (or prior probabilities) π_k for each component k. EM works by first choosing an arbitrary parameter set. In my implementation, I sampled the initial means from a zero-mean unit variance normal distribution, and the covariance matrices were initially set to the identity. I set π_1 to a random value between zero and one and set $\pi_2 = 1 - \pi_1$.

EM proceeds by estimating the "responsibilities" of each component for each data point in the E-step. The means, covariance, and mixing coefficients for both components are then re-estimated using the data weighted by the responsibility values in the M-step. This process alternates until a halting criterion is reached. I used the change in the log likelihood as my halting criterion and ran the algorithm until the change in log likelihood was less than $\epsilon = 0.005$.

2 Experiments

In order to assess the performance of EM on parameter evaluation for two-component Gaussian mixtures, I ran the algorithm 100 times, each time on a new set of 1000 points of randomly sampled data, and compared the results to the parameters of the Gaussian mixture from which the data was sampled. To keep the distributions relatively separable, I placed the means at $(2, 2)$ and $(-2, -2)$ and allowed the covariances to vary randomly. The covariance matrices were created by multiplying a normally distributed random matrix by its transpose to get a symmetric positive semi-definite matrix. This alone

	μ_{error}	π_{error}	misclass rate
KM	1.89	0.001	0.16
EM	0.49	0.022	0.06

Table 1: Comparison of k-means and EM averaged over 100 trials

produced many covariance matrices with relatively large off-diagonal terms (very long and skinny distributions), so I added an identity matrix to each Σ_k to make the distributions wider. I recorded the mean error, mixing coefficient error-both averaged across the two components-and the misclassification rate. As the covariance is not accounted for in k-means and a scalar error value for a matrix is not very informative without context, I did not report the covariance error. All measurements were averaged over all runs. The same values were measured for Matlab's default implementation of k-means (KM) as a benchmark for comparison with EM. Table 1 shows the results from this experiment.

Figure 1 shows the parameters of the Gaussian mixture converge on the solution over 120 iterations of EM for a particular run of the algorithm.

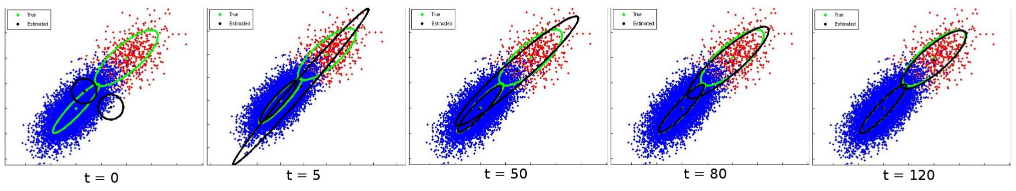

Figure 1: Progression of EM algorithm over 120 iterations

Figure 2 compares the correctly labeled distribution to the classification given by EM in the case where there is significant overlap between the two distributions.

3 Discussion and Conclusions

As may be expected, EM outperforms k-means in both mean error and classification rate. What is interesting was that I found that k-means consistently estimated the mixing coefficients better than EM in the few times I was able

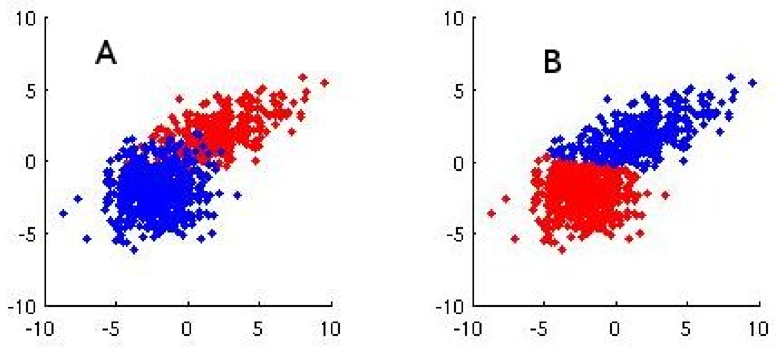

Figure 2: A shows the correctly labeled data from each component. B shows the labels given by the EM solution

to run the whole experiment. EM still performs qualitatively well, getting within 0.022 of the mixing coefficients, but it is surprising that with the naive assumptions of k-means (which don't take prior probabilities into account) that k-means would perform better on estimating the mixing coefficients.

Figure 1 shows a situation where EM nearly gets stuck in a local minimum and slowly climbs the likelihood surface. Most runs of EM on the random data sets took around 20 to 50 iterations to converge; this run took significantly longer. In situations like these randomized restarts or initialization using k-means might aid in getting to the solution more quickly.

Figure2 shows the limitations of EM when the two Gaussian distributions overlap. When categorizing a point there is a decision-theoretically optimal threshold which creates the hard cutoff in labeling where there is crossover in the original image.

For future work I would be interested in situations where it is not known how many Gaussians are in the mixture, or similarly, the data is not truly generated from a Gaussian mixture. Here model selection would be important and a method for scanning the solutions with different numbers of Gaussian components and choosing the simplest model that explains the data would likely be the crux of the problem. This could likely be approached using regularization or from a Bayesian perspective.