PR414 / PR813 Lecture 6 — Estimating models via the Expectation-Maximisation (EM) algorithm

This document is also available in PDF[1] format.

Purpose: To introduce the EM algorithm which is used to train mixture models, HMMs, etc.

Material: LECTURE NOTES[2] on GMMs, paper by TK Moon, "The Expectation Maximization Algorithm", IEEE Signal Processing Magazine, November 1996.

General: In this lecture we introduce a very powerful and elegant algorithm which can be used in situations where we have observations that are only indirectly tied to underlying parameters that we wish to estimate. It is extremely useful, has decent properties and find very wide application. Below we consider a number of applications of the EM algorithm, all of them directly relevant (there are more in the notes).

Some applications:

- **Clustering/Vector Quantisation (VQ):** Quite often in pattern recognition/coding, we observe a series of feature vectors $\boldsymbol{x}_1^T$ and want to find a series of representative vectors $\boldsymbol{c}_i$ (also called a "codebook") that can serve as a model/substitute for the original data $\boldsymbol{x}_1^T$. We want this codebook to be optimal in some sense, e.g. minimising the sum of squared quantisation errors. We have observed the feature vectors $\boldsymbol{x}_1^T$, but how can we estimate $\boldsymbol{c}_i$?
- **Mixture probability density functions (PDFs):** To escape the limitations of unimodality and symmetry of a Gaussian PDF we often prefer to substitute it with a mixture of Gaussian PDFs. If the match between a feature vector $\boldsymbol{x}$ and a single Gaussian PDF G_i is given by $f(\boldsymbol{x}|G_i)$, then its multimodal mixture version is given by $f(\boldsymbol{x}|M_i) = \sum_{k=0}^{K} w_k\, f(\boldsymbol{x}|G_k)$, where $\sum_{k=0}^{K} w_k = 1$. Once again, we have observed the feature vectors $\boldsymbol{x}_t$, but how can we use them to estimate the (unknown) underlying mixtures M_i? (By the way, the VQ problem is just a special case of mixture PDFs!)
- **Hidden Markov models (HMMs):** The scenario is very reminiscent of the above, only more complex. Now our model consists of a number of states $Q = j,\ j = 1..N$. Each state $i,\ i = 1..N$ has a transition probability a_{ij} joining it to state j. Each state i also has a PDF $f(\boldsymbol{x}|Q = i)$, which in its turn may be a mixture as described above. Typically we also have a probabilistic description of which states the model may start off in. How do we use feature vectors $\boldsymbol{x}_1^T$ to estimate these unknown parameters? (By the way, if you think about it long enough, you will discover that the mixture PDF problem is just a special (degenerate) case of the HMM!)

Description:

Before we continue on to the EM algorithm, we first need to introduce the concept of estimation, particularly **m**aximum **l**ikelihood **e**stimation (MLE). Consider a model with parameters Θ. We have a number of observations $\boldsymbol{x}_1^T$ from which we want to estimate Θ. The MLE approach is to consider the PDF $f(\boldsymbol{x}_1^T|\Theta)$ in a slightly unusual way. In contrast to normal use, we now take $\boldsymbol{x}_1^T$ as constant and Θ as a variable. When viewed from this perspective, we refer to the function value as a likelihood instead of a probability density value. We now search for the Θ so as to maximise the likelihood (ML) for the given $\boldsymbol{x}_1^T$. This value then serves as the ML estimate. When the relationship between the observations and the unknown parameter is direct, we can solve this equation via differentiation, numerical techniques or whatever. It is from this approach that we obtain the expressions we so often use for estimating means etc. In previous work where you estimated Gaussian PDFs the relationship

[1] http://www.dsp.sun.ac.za/pr813/lectures/6_em/6_em.pdf
[2] http://www.dsp.sun.ac.za/pr813/lectures/lecture06/pr813_lecture06.pdf

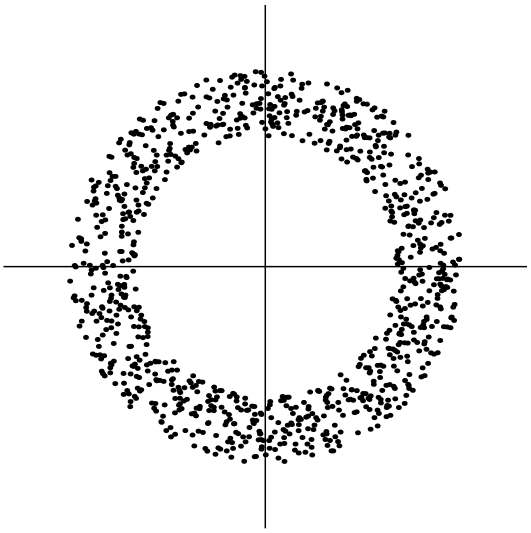
was direct and you (probably without knowing it) were using MLE. Please note, although MLE is very popular, it is by no means the only acceptable estimate. In later lectures we will investigate some alternatives. In the scenarios sketched above, notice that each time we have observations that are *only indirectly* related to the parameters that we want to estimate. For instance, in the Gaussian mixture case, we really would have wanted to know with which of the mixture components we must associate a given observation. In each case the problem was tricky because we were not sure where in the underlying model any given observation fitted. However, if we did have the underlying model available, we would probably be able to state where we *expect* a given observation to fit in. This is where the EM algorithm comes in.

- **Initialisation:** Assume some initial model. (The better you get it, the better you get it.)
- **Expectation Step:** Use you current model and the observed data to estimate the unknown factors that will make the relationship to the underlying model direct.
- **Maximisation Step:** Based on this (estimated) information now make a ML estimation of the parameters we are really interested in.
- **Convergence:** Iterate from the E-step until convergence.

The algorithm is guaranteed to converge (and normally does so fairly quickly), but it may only be a local optimum! The absence of tunable parameters is a major benefit. Compared to steepest descent and similar methods this is nirvana!

Project: (To hand in two lectures from today)

- (All) Play around with simple MLE to derive equations for estimating the (by now well-known) expressions for determining the parameters of Gaussian PDFs. (PR813 - why the N-1 term in the variance estimate?)
- (PR414 only full-covariance) Generate a doughnut-shaped *annulus* data set containing two-dimensional feature vectors, of which an example run is shown below.



Implement Gaussian mixture models based on full, diagonal and spherical covariance matrices, respectively. Fit each GMM to the annulus data set and display the resulting component densities on top of the data set, using ellipses centred at the component means. The shape of each ellipse is derived from the component covariance matrix `cv` via the following Matlab code snippet:

```
t = (0:200)*2*pi/200;
circle = [cos(t); sin(t)];
[U,D] = eig(cv);
ellipse = U*sqrt(D)*circle;
```

Discuss the differences between the three GMM versions, as well as the effect of the number of components K on the fit achieved.

- (All, PR414 only full covariance) Use Gaussian mixtures in the previous classification tasks (simvowel, speakers, faces). Compare results.