Annealing for Mixture Models*

Joseph Dickens
University of Michigan
josephdi@umich.edu

Roger Fan
University of Michigan
rogerfan@umich.edu

1 Introduction

Mixture models are a common and powerful method for clustering and distribution estimation. Data is modeled as a "mixture" of multiple simple distributions, most often Gaussian, where the goal is to estimate the parameters of each distribution in the mixture. In clustering problems we receive observations without knowing to which cluster they belong, a type of unsupervised learning. In this clustering setting, the alternative goal is often to estimate to which distribution each observation "belongs." Mixture membership can be modeled as a latent (unobserved) variable and the model evaluated under a likelihood framework. However, because the state space of this latent variable can be extremely large (for n data points from a k-component mixture there are k^n possible latent variable realizations), direct maximization of the likelihood is generally infeasible.

The most common method used to estimate parameters in this setting is the Expectation-Maximization (EM) algorithm [1]. The EM algorithm is an iterative optimization method used to optimize a lower bound on the likelihood. There are, however, several issues with using the EM algorithm even in this fairly simple case. Mixture models often have multiple local optima. Since EM is a hill-climbing algorithm, it may converge to a local optimum rather than a global optima [2]. This makes EM sensitive to the chosen starting point, and various ad-hoc methods (such as multiple start or random restart) are used in practice to attempt to mitigate this issue. EM can also be slow to converge in several scenarios, including when the mixing coefficients are unbalanced or there is significant overlap in the component distributions.

In this paper, we explore other methods that attempt to mitigate EM's issues by borrowing techniques from other optimization paradigms. Simulated annealing [3], a stochastic method designed to find global optima in large state-spaces, is of particular interest. There have been several previous attempts to apply annealing-inspired algorithms to mixture model estimation. Most of these methods tend to mimic EM, but replace the traditional "E-step" with a computationally attractive alternative. Such methods seek to explore the state space without directly hill-climbing, and possibly in a non-deterministic matter. The goal is to avoid local maxima and converge to a global maxima. The most notable algorithm is Deterministic Annealing EM (DAEM), proposed by Ueda and Nakano in 1998 [4]. As the name suggests, this algorithm replaces the stochastic nature of simulated annealing with a deterministic variant to hopefully avoid the local optima problem.

*This work was done as a final project for Stats 608A (Fall 2015) at the University of Michigan, taught by Professor Ambuj Tewari. The code used in this paper is available at https://github.com/rogerfan/stochasticEM and has been open-sourced under the Mozilla Public License, version 2.0.

We propose a new algorithm that we call Stochastic EM (SEM) and compare it to traditional EM and DAEM. We show that SEM is competitive with existing methods, and often exhibits better convergence rates on average. When the computing budget is high enough SEM can converge to significantly higher-likelihood states than EM or DAEM. However, due to its stochastic nature, SEM is not well-suited for all situations.

2 Previous Algorithms

2.1 Expectation-Maximization

In their paper first proposing the general EM Algorithm [1], Dempster et. al. demonstrate its applicability to mixture model estimation, and EM has become the standard method of estimating parametric finite mixture models.

Assume that data vectors $X_1, \ldots, X_n$ are i.i.d. draws from the J-component mixture distribution

$$P(X) = \sum_{j=1}^{J} \pi_j P_j(X \mid \theta_j)$$

Where π_j are the mixing components, P_j are the individual probability distributions, each indexed by a parameter vector θ_j. Note that we can reparameterize this model by introducing a latent variable Z that indicates group membership. Then our data generating model becomes:[1]

$$\begin{aligned} Z &\sim \text{Mult}(\pi_1, \ldots, \pi_J) \\ X \mid Z &\sim P_Z \end{aligned}$$

For simplicity, in this paper we will assume that the base distributions are multivariate Gaussian, so the parameters are $\theta_j = (\mu_j, \Sigma_j)$. Gaussian mixture models are by far the most commonly used in practice, but the algorithms and methods in this paper generalize to many base distributions.

To estimate this model, we can use the EM algorithm as follows:

1. Initialize parameters for each distribution $\mu_j^{(0)}$, $\Sigma_j^{(0)}$, and mixing components $\pi_j^{(0)}$.
2. Iterate the following steps until convergence or a computing budget is met:
   (a) E-step: Estimate the membership probabilities $p_{ij}^{(k+1)}$ for each observation i using the current parameter estimates $\mu_j^{(k)}$, $\Sigma_j^{(k)}$ and mixing components $\pi_j^{(k)}$.

$$p_{ij}^{(k+1)} = P(Z_i = j \mid X_i, \mu^{(k)}, \Sigma^{(k)}, \pi^{(k)}) = \frac{\pi_j^{(k)} P_j(X_i \mid \mu_j^{(k)}, \Sigma_j^{(k)})}{\sum_{j'=1}^{J} \pi_{j'}^{(k)} P_{j'}(X_i \mid \mu_{j'}^{(k)}, \Sigma_{j'}^{(k)})} \tag{1}$$

   (b) M-step: Estimate the parameters and mixing components using maximum likelihood estimation conditional on the current membership probabilities.

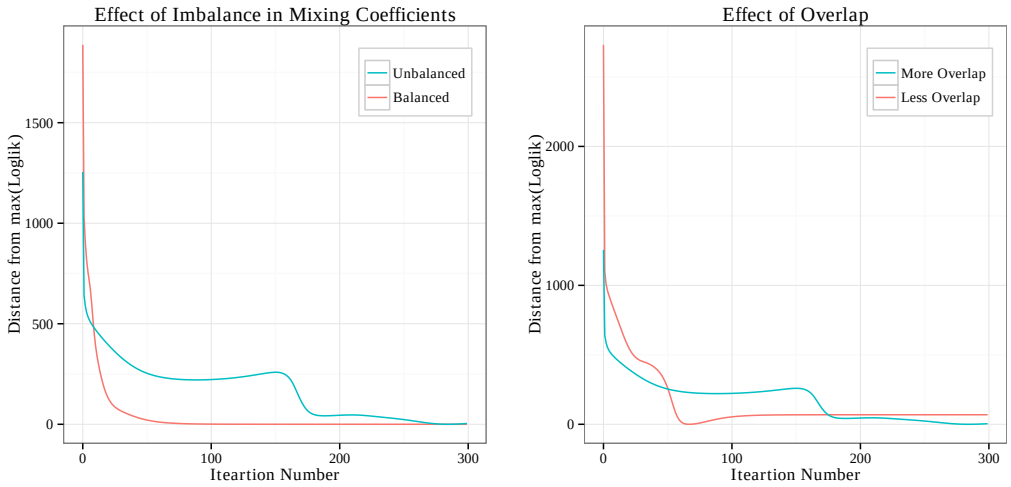
$$\begin{aligned} \pi_j^{(k+1)} &= \frac{1}{n} \sum_{i=1}^{n} p_{ij}^{(k+1)} \qquad \mu_j^{(k+1)} = \frac{\sum_{i=1}^{n} p_{ij}^{(k+1)} X_i}{\sum_{i=1}^{n} p_{ij}^{(k+1)}} \\ \Sigma_j^{(k+1)} &= \frac{\sum_{i=1}^{n} p_{ij}^{(k+1)} (X_i - \mu_j^{(k+1)})(X_i - \mu_j^{(k+1)})'}{\sum_{i=1}^{n} p_{ij}^{(k+1)}} \end{aligned} \tag{2}$$

[1] Note that Mult$(\cdot)$ here refers to the Multinomial distribution with $n = 1$ trials, otherwise known as the Categorical distribution.

2.2 Issues with EM

Despite its power and ubiquity, the EM algorithm is known to struggle in several common scenarios. Ueda and Nakano show that when clusters have significant overlap, convergence of EM tends to be much slower than when clusters are well separated in the data space [4]. Unfortunately, interesting problems are generally those with overlap, since unsupervised learning problems are much more difficult in these settings. Additionally, Naim and Gildea show that when mixing probabilities are unbalanced, i.e. when one or more classes have small mixing coefficients, EM can be very slow to converge as well [5].

Figure 1 demonstrates the effects of these issues with some simple comparisons. When the mixing components are unbalanced or there are significantly overlapping distributions, the EM algorithm tends to get stuck in slow-convergence areas, likely near saddle points. In the easier settings, the EM algorithm moves towards its limit much more quickly. In both of these cases the EM algorithm eventually escapes and converges to the maximum, but the convergence rate is severely effected.



(a) Unbalanced Mixing

(b) Overlapping Distributions

Figure 1: Situations where the EM algorithm has convergence issues.

2.3 Deterministic Annealing

Ueda and Nakano proposed a modification of the EM algorithm that they call Deterministic Annealing Expectation Maximization (DAEM) [4]. This algorithm is inspired by both simulated annealing and the EM algorithm, and introduces an inverse temperature parameter β that determines how "smoothed" the clusters are. The DAEM algorithm is the same as EM, except replacing Equation 1 (the E-step) with

$$p_{ij}^{(k+1)} = \frac{\left(\pi_j^{(k)} P_j(X_i \mid \mu_j^{(k)}, \Sigma_j^{(k)})\right)^{\beta_k}}{\sum_{j'=1}^{J} \left(\pi_{j'}^{(k)} P_{j'}(X_i \mid \mu_{j'}^{(k)}, \Sigma_{j'}^{(k)})\right)^{\beta_k}} \tag{3}$$

Note that when $\beta = 0$ the posterior membership probabilities become uniform, and when $\beta = 1$ we recover the standard EM algorithm. Generally we choose β_0 to be small, which does some smoothing between clusters, and allow $\beta_k \to 1$ as our iterations increase.

3 Simulated Annealing

Simulated annealing is a stochastic optimization algorithm designed to find global optima in optimization problems with large state spaces [3]. Consider the maximization problem

$$x^* = \arg\max_{x \in \Omega} f(x)$$

The general simulated annealing algorithm proceeds as follows:

1. Initialize at some point $x^{(0)}$ and initial temperature T_0.
2. Repeat until a computing budget is met:
   (a) Consider some stochastically chosen candidate point $x = g(x^{(k)})$.
   (b) If $f(x) > f(x^{(k)})$, accept the candidate move and set $x^{(k+1)} = x$.
   (c) Otherwise, accept the candidate move with probability $e^{(f(x)-f(x^{(k)}))/T_k}$.
   (d) If the candidate move is not accepted, then stay at the current point $x^{(k+1)} = x^{(k)}$.
   (e) Cool the temperature $T_{k+1} = \alpha T_k$ for some $0 \ll \alpha < 1$.

Note that the current temperature T_k determines how willing the algorithm is to accept moves that lower the objective function. For high temperatures, the algorithm accepts almost any move and uses this to explore the state space. As the temperature cools, the algorithm tends towards objective-improving moves and hopefully converges to the global optima.

In order to apply this algorithm to mixture models, we consider possible values of the latent mixture membership variable Z as the state space to optimize over. Given a realization of Z, we can write the log-likelihood function as

$$\log P(X) = \sum_{i=1}^{n} \sum_{j=1}^{J} \mathbb{1}(Z_i = j) \log P_j(X_i \mid \theta_j)$$

Plugging in the MLE estimates for θ_j conditional on Z, we therefore can apply simulated annealing to the objective function

$$f(Z) = \sum_{i=1}^{n} \sum_{j=1}^{J} \mathbb{1}(Z_i = j) \log P_j(X_i \mid \hat{\theta}_j) \tag{4}$$

3.1 Stochastic EM

We propose a new algorithm that we call Stochastic EM (SEM). This is an implementation of simulated annealing that randomly draws new realizations for the entire Z vector at each iteration. The SEM algorithm roughly follows the same structure as the EM algorithm:

1. Initialize parameters for each distribution $\mu_j^{(0)}$, $\Sigma_j^{(0)}$, mixing components $\pi_j^{(0)}$, and choose an initial temperature T_0.
2. Iterate the following steps until a computing budget is met:
   (a) Estimate the membership probabilities $p_{ij}^{(k+1)}$ using the current parameter estimates $\mu_j^{(k)}$, $\Sigma_j^{(k)}$ and mixing components $\pi_j^{(k)}$. Note that this is the exact same calculation as Equation 1, the E-step from the EM algorithm.

(b) Randomly draw candidate memberships $\tilde{Z} = (\tilde{Z}_1, \ldots, \tilde{Z}_n)'$, where

$$\tilde{Z}_i \sim \text{Mult}\left(p_{i1}^{(k+1)}, \ldots, p_{iJ}^{(k+1)}\right)$$

(c) Estimate candidate parameters and mixing components using maximum likelihood estimation conditional on the memberships $\tilde{Z}$. This is analagous to the M-step in EM (2).

$$\begin{aligned} \tilde{\pi}_j &= \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}(\tilde{Z}_i = j) \qquad \tilde{\mu}_j = \frac{\sum_{i=1}^{n} \mathbb{1}(\tilde{Z}_i = j) X_i}{\sum_{i=1}^{n} \mathbb{1}(\tilde{Z}_i = j)} \\ \tilde{\Sigma}_j &= \frac{\sum_{i=1}^{n} \mathbb{1}(\tilde{Z}_i = j)(X_i - \tilde{\mu}_j)(X_i - \tilde{\mu}_j)'}{\sum_{i=1}^{n} \mathbb{1}(\tilde{Z}_i = j)} \end{aligned} \tag{5}$$

(d) Calculate the candidate objective function

$$f(\tilde{Z}) = \sum_{i=1}^{n}\sum_{j=1}^{J} \mathbb{1}(\tilde{Z}_i = j) \log P_j(X_i \mid \tilde{\mu}_j, \tilde{\Sigma}_j)$$

(e) Update:

i. If $f(\tilde{Z}) > f(Z^{(k)})$ then accept the candidate memberships and update

$$\begin{aligned} Z^{(k+1)} &= \tilde{Z} & \pi^{(k+1)} &= \tilde{\pi} \\ \mu^{(k+1)} &= \tilde{\mu} & \Sigma^{(k+1)} &= \tilde{\Sigma} \end{aligned}$$

ii. Otherwise, acccept the candidate memberships with probability $e^{(f(x)-f(x^{(k)})/T_k}$.

iii. If the candidate move is not accepted, then stay at the current point $Z^{(k+1)} = Z^{(k)}$.

(f) Cool the temperature $T_{k+1} = \alpha T_k$.

3. Since the algorithm can accept worsening moves, return the function value and parameters associated with the best-so-far iteration $k^* = \arg\min_k f(Z^{(k)})$.

SEM is effectively a discrete, stochastic version of EM. Instead of using the membership probabilities, SEM only allows for discrete memberships and randomly chooses memberships at each iteration using the estimated probabilities. Due to this inaccuracy, we expect that the SEM will lose some convergence speed relative to standard EM, but hope that its discrete and stochastic nature will help it avoid local optima and saddle points.

4 Results

4.1 Data Generation

We simulate data from a 3-component Gaussian mixture model to compare these methods with. In order to provide a more challenging optimization environment, we include significant overlap and somewhat unbalanced mixing coefficients.[2] Figure 2 shows an example of $n = 2000$ draws from our mixture model. The majority of our analysis will be run on this data set.

[2]In particular, we set $\mu_1 = (0, 2)'$, $\mu_2 = (2, 1)'$, $\mu_3 = (2, 3)'$; $\Sigma_1 = \begin{pmatrix} 1 & 0.1 \\ 0.1 & 0.3 \end{pmatrix}$, $\Sigma_2 = \begin{pmatrix} 1 & -0.1 \\ -0.1 & 1 \end{pmatrix}$, $\Sigma_3 = \begin{pmatrix} 0.5 & -0.5 \\ -0. & 1 \end{pmatrix}$; and $\pi_1 = 0.15$, $\pi_2 = 0.70$, $\pi_3 = 0.15$.

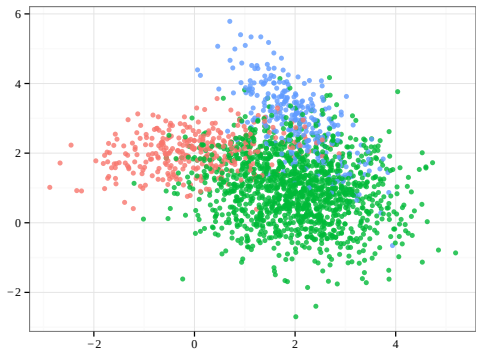
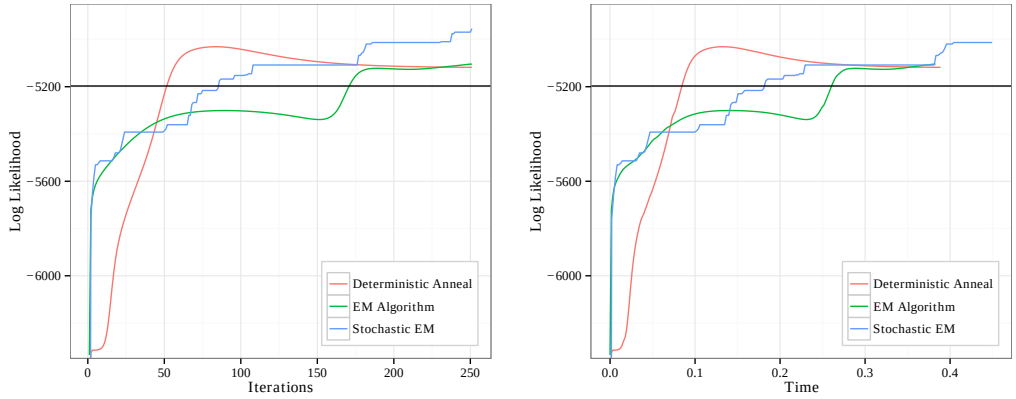


Figure 2: 2000 draws from our 3-component Gaussian mixture model.

4.2 Simulation results

Figure 3 shows the results of EM, DAEM, and SEM runs using the same initial parameter values for each algorithm.[3] We see that the EM algorithm exhibits the same issues described in Section 2.2, where the algorithm is caught in a slow-convergence area for a significant number of iterations. As expected, DAEM has slower initial convergence due to the smoothing of membership probabilities, but avoids slow-convergence areas and has better overall convergence performance.



(a) By Iterations

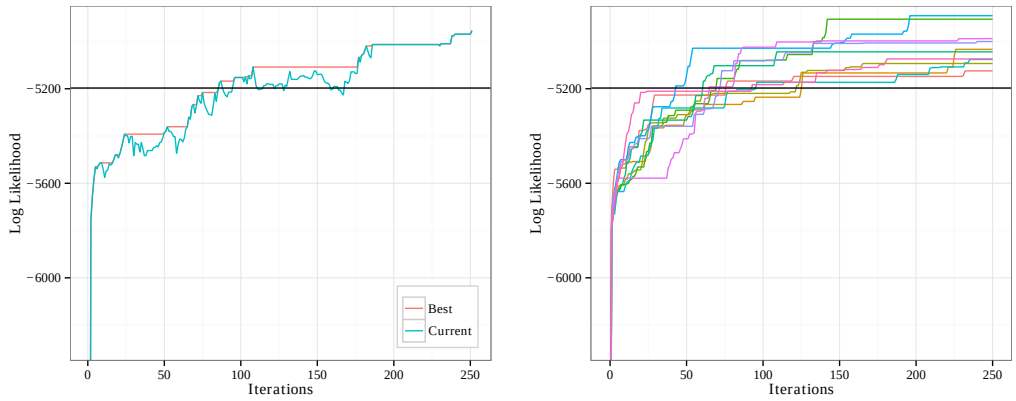
(b) By Time

Figure 3: Convergence performance of EM, DAEM, and SEM on the dataset shown in Figure 2, by both iteration number and processor time.

Stochastic EM seems to be fairly competitive with EM and DAEM, striking a middle point between the two. It attains the fast initial convergence of EM, but its stochastic nature allows it to escape the slow convergence area more quickly than regular EM. It also reaches a significantly higher final likelihood than either of the two other algorithms. Figure 3b shows how each of the algorithms performs against time. Stochastic EM

[3] SEM is estimated using an initial temperature of $T_0 = 100$ and a cooling schedule of $\alpha = 0.992$.

adds some computation to each step, but is still competitive with the other two methods. All three algorithms have the same order of computational complexity per iteration.



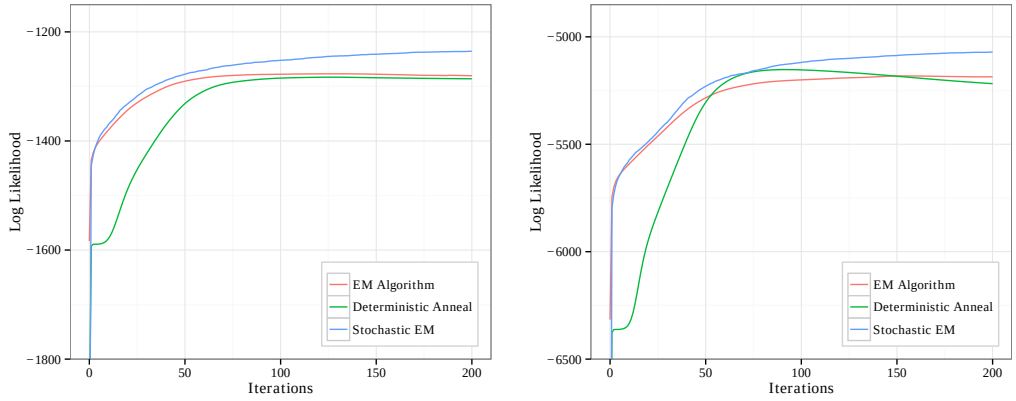
(a) Single SEM Run

(b) Multiple SEM Runs

Figure 4: SEM run examples. (a) compares the best-so-far log-likelihood to the log-likelihood of the current position. (b) shows multiple SEM runs from the same initial values.

Figure 4a shows how the likelihood of the current position (as opposed to the best-so-far) evolves in SEM. We can see that the algorithm's ability to accept objective-worsening moves allows it to further explore the state space and hopefully find better areas despite non-convexity.

We must keep in mind, however, that SEM is a stochastic algorithm. Figure 4b shows 10 example SEM runs on this data set, each using the exact same initial values. The final likelihood for even the worst-case SEM run is similar to those attained by EM and DAEM, but there is significant spread in both the convergence speed and final values. With enough computational time and slow enough cooling the algorithm should attain similar final results, but its stochastic nature means that it may not be well-suited for some applications.



(a) $n = 500$

(b) $n = 2000$

Figure 5: Average log-likelihood for each algorithm over 100 simulated datasets.

To further explore the effectiveness of SEM, we compared all three optimization methods on 100 simulated datasets for varying sample sizes. Figure 5 shows the average log-likelihoods of each method for these simulations. We can see that, on average, SEM is competitive with or outperforms both EM and DAEM. DAEM in particular seems to perform worse for smaller data sets, but has competitive performance with larger sample sizes. Once again, we see that SEM might overfit the data, since it converges to a higher final log-likelihood. We are primarily interested in whether the algorithms can recover the true mixture means.

5 Conclusion

Expectation-Maximization and EM-based algorithms are some of the most common methods for estimating mixture models. Existing deterministic methods, however, struggle to optimize in some conditions. The non-convex nature of mixture model likelihoods means that local optima or saddle points can stop or significantly slow down optimization progress. In this paper, we introduce a novel algorithm for estimating mixture models that we call Stochastic EM. This algorithm applies simulated annealing principles to the EM algorithm to hopefully avoid these issues and achieve more consistent global optimization. We implement SEM for Gaussian mixture models and show that it is competitive with existing algorithms and often outperforms them on-average. There are some tradeoffs, but SEM seems to be a promising alternative algorithm for estimating mixture models.

References

[1] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

[2] C. F. Jeff Wu. On the convergence properties of the EM algorithm. *The Annals of Statistics*, 11(1):95–103, 1983.

[3] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[4] Naonori Ueda and Ryohei Nakano. Deterministic annealing EM algorithm. *Neural Networks*, 11:271–282, 1998.

[5] Iftekhar Naim and Daniel Gildea. Convergence of the EM algorithm for Gaussian mixtures with unbalanced mixing coefficients. *Proceedings of the 29th International Conference on Machine Learning*, pages 1655–1662, 2012.