High-Dimensional Clustering with Sparse Gaussian Mixture Models

Akshay Krishnamurthy
Department of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
akshaykr@cs.cmu.edu

Abstract

We consider the problem of clustering high-dimensional data using Gaussian Mixture Models (GMMs) with unknown covariances. In this context, the Expectation-Maximization algorithm (EM), which is typically used to learn GMMs, fails to cluster the data accurately due to the large number of free parameters in the covariance matrices. We address this weakness by assuming that the mixture model consists of *sparse* gaussian distributions and leveraging this assumption in a novel algorithm for learning GMMs. Our approach incorporates the graphical lasso procedure for sparse covariance estimation into the EM algorithm for learning GMMs, and by encouraging sparsity, it avoids the problems faced by traditional GMMs. We guarantee convergence of our algorithm and show through experimentation that this procedure outperforms the traditional Expectation Maximization algorithm and other clustering algorithms in the high-dimensional clustering setting.

1 Introduction

Clustering is one of the most common machine learning tasks, and over the years, researchers have proposed numerous algorithms with not only impressive practical performance but also desirable theoretical guarantees. However, many of the most popular clustering algorithms, in particular, distance-based algorithms, perform poorly on high dimensional data because distances between points become more uniform [12]. Moreover, many of the approaches that address this concern, such as subspace clustering algorithms, make strong assumptions about the distributions of the data, and rely on heuristics to find clusters satisfying the assumptions [9]. Here, we present an alternative algorithm, based on the Gaussian Mixture Model, for high dimensional clustering that makes reasonable distributional assumptions and provides satisfying theoretical guarantees.

The Gaussian Mixture Model (GMM) is a generative approach to clustering, where each cluster corresponds to a Gaussian distribution, and each example is drawn from its cluster-specific distribution. Given a set of examples, one can use the Expectation-Maximization algorithm to learn the parameters of the model that maximize the likelihood of the data [1]. Once estimated, these parameters naturally lead to a clustering of the data: assign each example to the Gaussian distribution it is most likely to originate from. Parameter estimation in GMMs is a maximum likelihood estimation problem, and therefore, clustering benefits from the statistical properties of these estimators. Additionally, it is well known that GMMs work quite well in practice, especially on low-dimensional problems.

Unfortunately, GMMs, like other clustering algorithms, experience severe performance degradation in the high-dimensional setting. This failure occurs because there are enough free parameters in the covariance matrices of the cluster distributions to provide a good fit of many different assignments of examples to clusters. Thus, in the learning stage, the Expectation-Maximization algorithm typically converges to an incorrect set of parameters for the Gaussians, and this results in poor clustering performance. This is fundamentally an overfitting issue, and it arises due to the increase in the number of parameters when in high dimension.

A popular technique for overcoming the curse of dimensionality and to prevent overfitting is to regularize, or constrain, the parameters of the model. One popular type of regularization, ℓ_1 regularization, encourages many of the parameters to be 0, and typically results in sparse models that generalize well. An ℓ_1 (or Lasso) regularized solution provides a simple explanation for the data. This is natural as we expect that data arises from simple processes, in accordance with Occam's Razor. Lasso regularization has been applied with incredible success to both regression and classification problems (see [5, 13]), and recently, it has also been used for model selection, particularly in Gaussian graphical models (multivariate Gaussian distributions). The Graphical Lasso is one optimization procedure that efficiently learns sparse inverse covariance matrices, which correspond to distributions where only a few pairs of variables are dependent conditioned on all of the other variables [3]. Encouraging sparsity in the inverse covariance matrix results in a simple model that still fits the data well.

Given the recent work on learning sparse gaussian graphical models, it seems natural to constrain the cluster distributions in GMMs to also be sparse, and this is the approach we take in this paper. We augment the Expectation-Maximization algorithm for learning GMMs to encourage sparsity in the cluster distributions. We show that this procedure corresponds to maximizing a likelihood function with a prior placed on each inverse covariance matrix, and thus guarantee convergence of our algorithm. In addition, we study the empirical performance of this approach, and show that it outperforms GMMs and other clustering techniques in the high-dimensional setting.

This paper is organized as follows: in Section 2 we review some existing algorithms for high-dimensional clustering. In Section 3, we formalize the clustering problem and present the Gaussian Mixture Model and the Graphical Lasso. Then, we combine these two procedures and present our algorithm for learning Sparse GMMs (sGMMs) in Section 4. We also provide proofs of numerical convergence and guarantees for our algorithm. In Section 5, we present several experiments that demonstrate the practical performance of sGMMs on simulated and real data. We conclude in Section 6 with a brief discussion and some ideas for future work.

2 Related Work

One common approach to high-dimensional clustering is to first perform dimensionality reduction as a pre-processing step and then cluster in the reduced-dimension space. One obvious approach is to perform Principal Component Analysis to project the data into a low-dimensional subspace that captures most of the variance in the data. Constructing the similarity matrix using Euclidean distances in this subspace typically results in more accurate clustering. Of course, if the clusters are non-linear in the ambient space, this algorithm will perform poorly. For example, in [14], Yeung et al. conclude that PCA as a preprocessing step is not suitable for clustering gene expression data.

Spectral clustering is an extremely popular technique that addresses the concerns of PCA by accomodating clusters that lie on non-linear manifolds [6]. In this algorithm, the similarity matrix is computed using a kernel function, typically the Gaussian Kernel ($K(x_i, x_j) = \exp\{-||x_i - x_j||^2/\sigma\}$ for some parameter σ), and the largest eigenvectors of the Laplacian of this matrix are used to cluster the data points. Spectral clustering performs quite well in practice, but we believe that in some cases, taking advantage of the cluster structure can outperform this algorithm. This motivates our use of Gaussian Mixture Models.

Another related technique is Subspace Clustering, which attempts to find small subspaces of the ambient space that captures most of the variance in the data and clusters the data in these subspaces [7, 9]. This class of algorithms is motivated by the fact that PCA finds a projection that maximizes the variance of all of the data, but for clustering, we are actually interested in subspaces that maximizes between cluster distance and minimize within cluster distances. In general, these two criteria are not equivalent, and thus PCA often does not improve clustering performance.

In theory, subspace clustering looks to find clusters in each of the different subspaces, and it evaluates each clustering with some score criterion. The output of the algorithm is the clustering that maximizes (or minimizes) the score. One common score function is the entropy of the subspace, which is based on the intuition that a good clustering has low entropy. Of course, there are exponentially many subspaces making an exhaustive search intractable, and the many variants of subspace clustering differ in their heuristic algorithms for exploring the subspaces (See [9] for a survey of some of these algorithms). The challenge of searching for the best subspaces makes these methods undesireable, so it is still important to consider alternative algorithms.

One reasonable alternative to subspace clustering algorithms is the Gaussian Mixture Model, which is a well known generative model that is commonly used for clustering. Unfortunately, in high dimensional settings, the limited number of samples results in non-singular covariance matrices, and thus some form of regularization is required. In his thesis, Elliot considers two main forms of covariance regularization for GMMs, one of which is a shrinkage estimator where $\Sigma = \lambda T + (1 - \lambda)\hat{\Sigma}$ for some target matrix T [2]. He performs experiments on an image dataset and concludes that the shrinkage estimator where T is the identity matrix outperforms his other methods. This shrinkage estimator is undesireable as it fails to capture the complexity of each cluster. However, some form of regularization is required; this motivates our use of ℓ_1 regularization for learning GMMs.

Finally, Pan and Shen consider a regularized Gaussian Mixture Model, but they only seek to find sparse mean vectors [8]. Their algorithm uses a shared identity covariance matrix and as a result does not address the same problem as our methods.

3 Background

Suppose we are given $\{X_1^{(i)}, ..., X_p^{(i)}\}_{i=1}^n$, where each $X^{(i)} \in \mathcal{R}^p$. The goal of clustering is to find a function $\hat{f} : \mathcal{R}^p \to \{1, ..., k\}$ that associates each data point with one of k clusters. We are interested in finding $\hat{f}$ using the Gaussian Mixture Model, which learns k Gaussian distributions (with means μ_j and covariances Σ_j) and uses $\hat{f}_{GMM}(X_i) = \text{argmax}_{j \in \{1,...,k\}} \phi_{\mu_j, \Sigma_j}(X_i)$, where $\phi_{\mu,\Sigma}$ is the Gaussian density with mean μ and covariance Σ. The GMM is a generative model where each sample is generated by first choosing a cluster and then drawing from that cluster's Gaussian distribution. Parameters of the mixture model are learned by maximizing the incomplete data likelihood:

$$\mathcal{L}(\Theta; X^{(1)}, .., X^{(n)}) = \prod_{i=1}^{n} \sum_{j=1}^{k} \mathbb{P}(Y^{(i)} = j)\mathbb{P}(X^{(i)} | Y^{(i)} = j, \Theta); \tag{1}$$

Where $Y^{(i)}$ represents the cluster that produced $X^{(i)}$. The distribution of each $Y^{(i)}$ is a multinomial with parameter vector $(\pi_1, ..., \pi_k)$, and the distribution $X^{(i)}|Y^{(i)} = j$ is a Gaussian with mean μ_j and covariance Σ_j. The parameter vector Θ consists of the multinomial parameters, as well as all of the Gaussian means and covariances. Exact parameter estimation in the GMM is intractable since we do not know which Gaussian generated each $X^{(i)}$ and we have to marginalize over $Y^{(i)}$. Instead, a popular approach is to use the Expectation-Maximization algorithm (EM), where in the E-step, we compute the probability that each $X^{(i)}$ originated from each Gaussian, and in the M-step, we perform maximum likelihood estimation to update the μ_js and Σ_js using the data, weighted by the probabilities computed in the E-step. This iterative algorithm will find a local optimum of the likelihood function, but since this objective is non-convex, there are no guarantees of finding a global optimum.

A related problem is the covariance estimation problem for Gaussian graphical models. It is well known that in high dimension, the MLE for the covariance matrix is prone to overfitting, and moreover if the number of samples n is smaller than the dimension p, the MLE is non-singular and an unsuitable estimate. A popular alternative that overcomes both of these issues involves placing an ℓ_1 penalty on the entries of the inverse covariance matrix (the concentration matrix). Zeros in this matrix correspond to conditional independencies between random variables. Given data $\{X_1^{(i)}, ..., X_p^{(i)}\}_{i=1}^n$ from a Gaussian distribution, maximizing the negative penalized log-likelihood results in the following optimization:

$$\hat{C} = \text{argmin}_C - \log|C| + \text{tr}(C\hat{S}) + \lambda||C||_1 \tag{2}$$

Algorithm 1 Train sGMM($\{X_1^{(i)}, ..., X_p^{(i)}\}_{i=1}^n$, λ, k)

```
Initialize $\pi_1, ..., \pi_k, \mu_1, ..., \mu_k, \Sigma_1, ..., \Sigma_k$ randomly
repeat
  for $j \in \{1, ..., k\}$ do
    Compute $p_{i,j} \propto \pi_j \phi(X^{(i)} | \mu_j, \Sigma_j)$.
    $\mu_j \leftarrow \frac{\sum_{i=1}^n p_{i,j} X^{(i)}}{\sum_{i=1}^n p_{i,j}}$ and $S_j \leftarrow \frac{\sum_{i=1}^n p_{i,j}(X^{(i)} - \mu_j)^T (X^{(i)} - \mu_j)}{\sum_{i=1}^n p_{i,j}}$
    $C_j \leftarrow \text{argmin}_C - \log |C| + \text{tr}(CS_j) + \lambda ||C||_1$
    $\Sigma_j \leftarrow C_j^{-1}$
    $\pi_j \leftarrow \frac{1}{n} \sum_{i=1}^n p_{i,j}$
  end for
until convergence
```

Where C is the concentration matrix with $C = (c_{i,j})$, $\hat{S} = \sum_{i=1}^n (X_i - \mu)$ is the sample covariance matrix, and $||C||_1 = \sum_{i,j} |c_{i,j}|$ is the L_1 norm of C. This function is convex, and there are several procedures for solving this problem (see for example [3, 15]). One algorithm involves iteratively solving coupled lasso regression problems until convergence [3]. Additionally, Yuan and Lin show that asymptotically, $\hat{C}$ selects the right graphical model structure and is a consistent estimator of C, the true concentration matrix [15].

4 Methods

We are interested in clustering data, $\{X_1^{(i)}, ..., X_p^{(i)}\}_{i=1}^n$, when p is large. We assume that there are k latent gaussian distributions, with means μ_j and covariances Σ_j, just as in the GMM problem formulation. We also assume that each sample $X^{(i)}$ was produced from the following generative model: Choose $j \in \{1, ..., k\}$ and draw $X^{(i)} \sim \mathcal{N}(\mu_j, \Sigma_j)$. The goal of our algorithm is to learn the parameters μ_j, Σ_j as well as the mixture proportions, which are the probabilities of choosing a specific cluster j in the generative process.

The problem with learning gaussian distributions in high dimension is that $\Sigma_j \in \mathcal{R}^{p \times p}$, so the number of parameters scales with p^2. We typically do not have enough data to robustly learn all of these parameters, and this leads to overfitting. In the context of GMMs, the learning algorithm typically converges to a random assignment of points to clusters, because we can fit any such assignment reasonably well, and this effect severely degrades the performance of GMMs for clustering. One well known technique for addressing the problem of overfitting in gaussian graphical models (and in general), is to penalize complex solutions. Our algorithm does this by encouraging sparse concentration matrices with a ℓ_1 penalty on individual entries in these matrices. We call this algorithm sGMM for sparse Gaussian Mixture Model.

We consider the following objective function, which is the incomplete penalized log likelihood:

$$\mathcal{L}(\Theta; X^{(1)}, ..., X^{(n)}) = \sum_{i=1}^n \log \left(\sum_{j=1}^k P(Y^{(i)} = j) \mathbb{P}(X^{(i)} | Y^{(i)} = j, \Theta) \right) + \sum_{j=1}^k \log \mathbb{P}(C_j) \quad (3)$$

This objective is quite similar to the objective for GMMs (Equation 1), except for the term involving the concentration matrices. If each $c_{i,j} \sim Laplace(0, 1/\lambda)$, then this term becomes $\sum_{j=1}^k \lambda ||C_j||_1$, which is exactly the ℓ_1 penalty in Equation 2. Thus, introducing the ℓ_1 penalty can be viewed as performing maximum a posteriori inference, with Laplace priors on the concentration matrices.

As exact maximization of Equation 3 is intractable, we use the Expectation-Maximization algorithm to iteratively find a local optima. The E-step of this procedure is identical to the E-step for training GMMs; we simply computing the probability that each training point was drawn from each of the k Gaussian distributions. The M-step differs in that we run the graphical lasso procedure (See [3]) for each cluster, using the weighted sample variance as input. See Algorithm 1 for details.

Theorem 4.1. *Algorithm 1 finds a set of parameters Θ that achieves a local optima of Equation 3.*

Proof. The first term in Equation 3 is exactly the objective for traditional Gaussian Mixture Models. We can bound this term from below by $\sum_{i=1}^{n} \mathbb{P}(Y^{(i)} = j|X^{(i)}, \Theta) \log \mathbb{P}(X^{(i)}|Y^{(i)} = j, \Theta)$ using Jensen's inequality. This gives us a lower bound on the objective in Equation 3 and Algorithm 1 is an instance of the EM algorithm that is exactly maximizing this lower bound objective using a form of block-coordinate ascent.

In the E-step, we fix the parameter values and find a distribution Q that is closest in KL-divergence to $\mathbb{P}(\mathbf{Y}|\mathbf{X}, \Theta^{(t)})$. Since the parameter values are fixed in this step, we can ignore the second term in Equation 3. Using the fact that the first term is the GMM objective, we know that the distribution defined by the $p_{i,j}$s minimizes KL-divergence to the conditional. Therefore, the updates in th E-step are guaranteed to not lower the objective. The M-step is simply maximum likelihood estimation of the full objective, and we know that this will also not decrease the objective. These two properties imply that the algorithm will converge to a local optima. □

With Theorem 4.1, we at least have some guarantee on the performance of the sGMM. In addition to this guarantee, our experiments in Section 5 show that when the assumptions of this model are satisfied (i.e. data is drawn from Gaussian distributions), this algorithm performs quite well, often outperforming state-of-the-art clustering algorithms.

4.1 Hyperparameter Selection

Algorithm 1 requires two hyperparameters, namely the regularization parameter λ and the number of clusters k. As is true with other ℓ_1 regularization methods, it is critical to choose λ correctly, as this parameter can significantly affect the accuracy of the learned model. We address this concern with cross validation, where we train sGMMs on a fraction of the data, and evaluate the likelihood of each model on the rest of the data set. We then choose the regularization parameter that maximizes the likelihood on the held-out data.

A more general concern with Algorithm 1, and many clustering algorithms, is that we must know a priori the number of clusters in our data, but this information is typically unavailable. This problem cannot be addressed with cross validation, as allowing for more clusters will always provide a better fit and increase the likelihood on the held-out data set. A common technique is to perform model selection using the AIC penalty, which chooses the model that maximizes $\ell(\Theta_k; X^{(1)}, ..., X^{(n)}) - |\Theta_k|/2$ where $|\Theta_k|$ is the number of parameters in that model. For sGMMs, $|\Theta_k| = kp^2/2 + kp/2 + k - 1$ grows with the square of the number of dimensions, and for high-dimensional problems, this penalty is too severe. Our experiments in Section 5 show that using the AIC penalty consistently finds models with too few clusters and this results in poor clustering performance. Thus, this form of model selection is also not suitable for our purposes.

5 Experiments

To assess the performance of our algorithm, we compare to three other clustering algorithms, namely traditional GMMs with a covariance matrix shared by all clusters, spectral graph clustering with the Gaussian kernel, and K-Means clustering with Euclidean distances. For these experiments, we generated data for 2 gaussian distributions with varying mean distances and with the following concentration matrices from Yuan and Lin [15]: *AR1* with $c_{i,i} = 1$ and $c_{i,i-1} = c_{i-1,i} = 0.5$; *AR2* with $c_{i,i} = 1$, $c_{i,i-1} = c_{i-1,i} = 0.5$ and $c_{i,i-2} = c_{i-2,i} = 0.25$; *AR4* with $c_{i,i} = 1$, $c_{i-1,i} = 0.4$, $c_{i-2,i} = c_{i-3,i} = 0.2$, and $c_{i-4,i} = 0.1$; and *FULL* - $c_{i,i} = 2$ and $c_{i,j} = 1$ otherwise.

Figure 1(a) shows the performance of the four clustering algorithms as a function of dimension. 100 samples was generated from each of two AR1 gaussian distributions, one centered at the origin, and the other centered at $(3/\sqrt{p}, ..., 3/\sqrt{p})$. Training error was computed by finding a permutation of the labels output by the clustering algorithm that matched the true labels of the samples. Figure 1(b) shows the results for the same experiment on two FULL gaussian distributions, and Figure 1(c) shows the same experiment where one gaussian is from the AR1 model and the other is from the FULL model. In these experiments, cluster means are purposely quite close together, simulating a clustering problem that is challenging for all algorithms. Additionally, for all of these experiments,

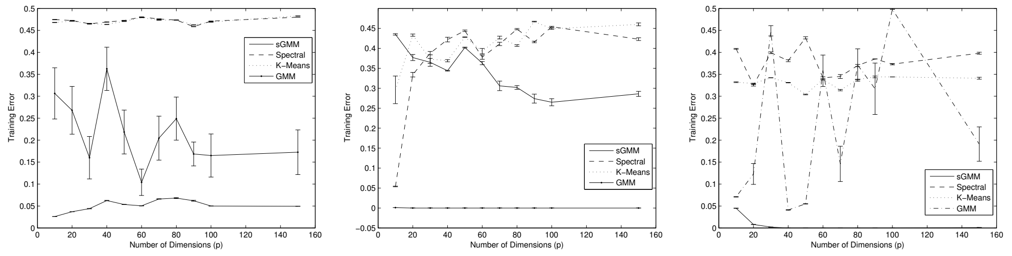

Figure 1: Experiments on simulated data. (a): Training error as we increase p, the number of features on the AR1 model. (b): On the FULL model. (c) On two gaussians, one from the FULL model and one from the AR1 model.

we chose $\lambda = 0.3$, which we experimentally found to be a reasonable choice in the absence of cross validation.

These experiments show that in high-dimension sGMM typically outperforms state-of-the-art clustering algorithms even when the sparsity assumption is violated, as in Figures 1(b) and 1(c). One interesting observation from the AR1 experiment is the high variance of traditional GMMs in comparison with sGMM and the other cluster algorithms. We believe this happens because the algorithm for training GMMs is heavily influenced by initial choices for the parameters. Given the low variance of sGMM, it seems that imposing sparsity reduces this effect. Another observation is that GMMs consistently achieved zero training error when data was generated from two FULL gaussians. This is not unreasonable as the two clusters share the same covariance matrix, so we would expect the unsparse mixture model to perform well, but the result is somewhat surprising. As shown in Figure 1(c), when the clusters do not share the same covariance matrix, the performance of the standard GMM is again highly variable, whereas sGMM performs extremely well.

Additionally, we studied how the distance between clusters and the number of samples affects performance. For the former, we evaluated the training error as a function of distance between cluster means for the 4 algorithms on the AR1 model. We held p fixed at 100 dimensions, and drew $n = 100$ samples from each distribution. Results for this experiment are in Figure 2(a). While all three algorithms quickly drop to zero training error, we found that sGMM is much better at finding clusters that are close together. For the latter experiment, we again computed training error as a function of the number of samples n for each of the four algorithms. Surprisingly, we found that the number of samples had little effect on training error, and consequently, these experiments are not shown.

To motivate the need for cross validation, we also studied the effect of the regularization parameter λ on clustering performance. Figure 2(b) traces this effect for three of the gaussians described above. Again for this experiment, we fixed $p = 100$, $d = 3$, and $n = 500$. The results show that, as expected, the choice of λ can have a substantial effect on the final accuracy of the algorithm, but also that $\lambda = 0.3$ could be a reasonable choice for many models. Finally, we analyzed how using AIC for model selection affects clustering performance. We generated data from k clusters and used the model selection procedure describe in Section 4.1 and computed the training error of the best model. Results for this experiment for various values of p are in Figure 2(c). As mentioned in Section 4.1, we noticed that the AIC penalty is too severe of a penalty in high dimension, so a model with few clusters (typically just one or two) is chosen, and this results in poor clustering performance.

5.1 MNIST Digit Dataset

Finally, we analyzed the performance of sGMM on the MNIST handwritten digit dataset [4]. This data set consists of hundreds of sample handwritten digits (10 clusters), each represented as an 8×8 matrix of integers. For our experiments, we ran our clustering algorithms on a data set of 100 samples drawn randomly from each cluster. In Figure 3(a), we plot training error as a function of λ when we ran the clustering algorithms on data drawn from all 10 clusters. In Figure 3(b), we only look at four clusters in the data set, specifically the clusters corresponding to zero, one, seven, and eight.

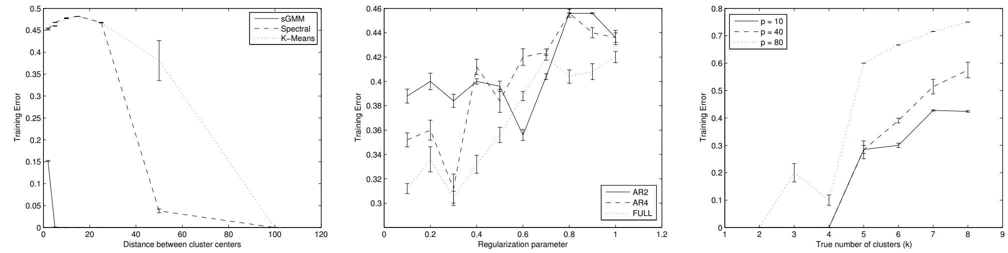

Figure 2: Additional experiments on simulated data. (a): Training error as a function of distance between cluster centers on the AR1 model ($p = 100, k = 2, n = 200$). (b): Generalization error as a function of regularization parameter λ for 3 different gaussian distributions ($p = 50$, $k = 2$). (c) Training error when the number of clusters k is unknown, using AIC model selection, for 3 different values of p.

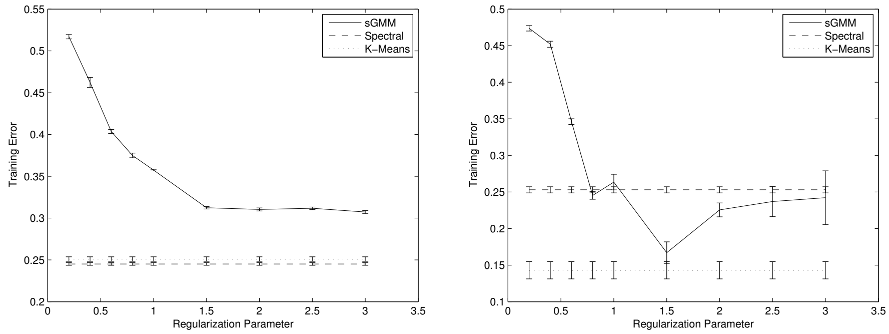

Figure 3: Experiments on MNIST handwritten digit dataset. (a): Training error for spectral clustering, k-means, and sGMM on all 10 clusters. (b): Training error for the three algorithms on only 4 of the clusters.

On both datasets, we found that sGMM performs worse than spectral clustering, and surprisingly, K-Means performs better than both algortihms. There are two possible explanations for this: one is that the assumptions of sparsity and gaussianity may be violated by that data, and the other is that sGMM might be more suitable for clustering problems where there are only two or three clusters in the data. If the former is true, there is no reason to expect that sGMM performs well. However, the latter explanation may be more reasonable, and indeed, we found that when we only used data from two clusters, sGMM typically achieved much lower training error.

6 Conclusion

In this paper, we propose sparse Gaussian Mixture Models, a novel algorithm for high-dimensional clustering, which relies on ℓ_1 penalized concentration matrices in the standard mixture model framework. Using the fact that the regularization is equivalent to placing a prior on the parameters in the model, we prove that our training algorithm (Algorithm 1) is guaranteed to find a local maxima of our objective (Equation 3). We also consider practical issues with this algorithm, including cross validation for hyper-parameter selection and model selection to choose the number of clusters.

We perform several experiments to study the performance of our methods on real and simulated data. At a high level, these experiments show that when the assumptions, namely sparsity and gaussianity, of our model are satisfied sGMMs performs quite well in practice, often outperforming state-of-the-art algorithms like spectral clustering. We assess how hyperparameter selection affects clustering performance, and conclude that cross-validation is a necessary aspect of training sGMMs. We show that model selection using AIC severely penalizes models with many clusters and therefore is not suitable for our purposes. Finally, we evaluated our algorithms on the MNIST handwritten digit dataset but found that sGMMs typically performed worse that spectral clustering and K-Means.

One direction for future work is to explore alternatives to AIC for model selection. Since we consider a generative process, one option is to consider an infinite mixture model, specifically the Infinite Gaussian Mixture model [10]. In this model, a Dirichlet Process is used to generate the multinomial parameters π, allowing for the number of clusters to be chosen in a data-driven fashion. Of course inference in this model is challenging, but several sampling methods perform well in practice. An interesting question is whether enforcing sparsity in the infinite mixture model improves its ability to find clusters within the data.

Another alternative is a variant of the AIC that does not over-penalize sparse solutions. In [11], Städler at all consider a problem similar to the sparse gaussian mixture model, and they use a BIC penalty where they only count the number of non-zero parameters (the effective number of parameters). The intuition is that we should not penalize the model for the parameters that are set to zero, as they do not add to model complexity. This idea is also used by Pan and Shen in [8]. This penalty, or a variant of it, may be a better option for model selection in the sparse GMM, and we hope to experiment with this in the future.

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*, 39(1):1–38, 1977.

[2] Daniel L Elliot. *Covariance Regularization in Mixture of Gaussians for High-Dimensional Image Classification*. PhD thesis, Colorado State University, 2009.

[3] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics (Oxford, England)*, 9(3):432–41, July 2008.

[4] Yann Lecun and Corinna Cortes. The MNIST database of handwritten digits, 1998.

[5] Su-in Lee, Honglak Lee, Pieter Abbeel, and Andrew Y Ng. Efficient L1 Regularized Logistic Regression. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence (AAAI)*, 2006.

[6] Ulrike Luxburg. A Tutorial on Spectral Clustering. *Statistics and Computing*, 17(4):395–416, August 2007.

[7] Andrew McCallum, Kamal Nigam, and Lyle H. Ungar. Efficient clustering of high-dimensional data sets with application to reference matching. *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '00*, pages 169–178, 2000.

[8] Wei Pan and Xiaotong Shen. Penalized Model-Based Clustering with Application to Variable Selection. *Journal of Machine Learning Research*, 8:1145–1164, 2007.

[9] Lance Parsons, Ehteshan Haque, and Huan Liu. Evaluating Subspace Clustering Algorithms. In *SIAM International Conference on Data Mining*, 2004.

[10] Carl Edward Rasmussen. The Infinite Gaussian Mixture Model. In *Advances in Neural Information Processing Systems*, pages 554–560, 2000.

[11] Nicolas Städler, Peter Bühlmann, and Sara Van De Geer. \ell_1-Penalization for Mixture Regression Models. *Submitted*, pages 1–35, 2010.

[12] Michael Steinbach, Lievent Ertöz, and Vipin Kumar. The Challenges of Clustering High Dimensional Data. In *In New Vistas in Statistical Physics: Applications in Econophysics, Bioinformatics, and Pattern Recognition*, pages 1–33, 2003.

[13] Robert Tibshirani. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society*, 58(1):267–288, 1996.

[14] K Y Yeung and W L Ruzzo. Principal Component Analysis for Clustering Gene Expression Data. *Bioinformatics*, 17(9):763–774, 2001.

[15] M. Yuan and Y. Lin. Model selection and estimation in the Gaussian graphical model. *Biometrika*, 94(1):19–35, February 2007.