ABSTRACT

Title of dissertation: GLOBAL OPTIMIZATION OF FINITE MIXTURE MODELS

Jeffrey W. Heath
Doctor of Philosophy, 2007

Dissertation directed by: Professor Michael Fu
Robert H. Smith School of Business
&
Professor Wolfgang Jank
Robert H. Smith School of Business

The Expectation-Maximization (EM) algorithm is a popular and convenient tool for the estimation of Gaussian mixture models and its natural extension, model-based clustering. However, while the algorithm is convenient to implement and numerically very stable, it only produces solutions that are locally optimal. Thus, EM may not achieve the globally optimal solution in Gaussian mixture analysis problems, which can have a large number of local optima. This dissertation introduces several new algorithms designed to produce globally optimal solutions for Gaussian mixture models. The building blocks for these algorithms are methods from the operations research literature, namely the Cross-Entropy (CE) method and Model Reference Adaptive Search (MRAS).

The new algorithms we propose must efficiently simulate positive definite covariance matrices of the Gaussian mixture components. We propose several new solutions to this problem. One solution is to blend the updating procedure of CE and

MRAS with the principles of Expectation-Maximization updating for the covariance matrices, leading to two new algorithms, CE-EM and MRAS-EM. We also propose two additional algorithms, CE-CD and MRAS-CD, which rely on the Cholesky decomposition to construct the random covariance matrices. Numerical experiments illustrate the effectiveness of the proposed algorithms in finding global optima where the classical EM fails to do so. We find that although a single run of the new algorithms may be slower than EM, they have the potential of producing significantly better global solutions to the model-based clustering problem. We also show that the global optimum matters in the sense that it significantly improves the clustering task.

Furthermore, we provide a a theoretical proof of global convergence to the optimal solution of the likelihood function of Gaussian mixtures for one of the algorithms, namely MRAS-CD. This offers support that the algorithm is not merely an ad-hoc heuristic, but is systematically designed to produce global solutions to Gaussian mixture models. Finally, we investigate the fitness landscape of Gaussian mixture models and give evidence for why this is a difficult global optimization problem. We discuss different metrics that can be used to evaluate the difficulty of global optimization problems, and then apply them to the context of Gaussian mixture models.

GLOBAL OPTIMIZATION OF FINITE MIXTURE MODELS

by

Jeffrey W. Heath

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2007

Acknowledgements

I owe tremendous gratitude to many people for helping me along my graduate career and making this dissertation possible. First, I would like to thank my wife Shanna for her endless support of everything I do. She never ceases to be my biggest fan. To my father Wells, my mother Malissa, and my sister Allison, thank you for being there every step of the way and for always being proud of me. The support I received from my grandparents, aunts, uncles, cousins, as well as Shanna's family served as great inspiration throughout my education. I am truly blessed to be part of such a wonderful family.

I cannot thank my two advisors, Professor Michael Fu and Professor Wolfgang Jank, enough for their countless hours of guidance and collaborations over the last few years. I was honored to work with not one, but two amazing professors who not only excel in their research, but genuinely care deeply for their students. Thanks to Professor Fu and Professor Jank for the patience, kindness, and relentless encouragement you showed me in all of our weekly meetings. Your ideas, discussions, and positive attitudes always made me more excited about my research. Thanks as well to Professors Bruce Golden, Paul Smith, and Gurdip Bakshi for serving on my dissertation committee. I would also like all of the teachers who have helped me in my academic career over the years, notably Christine Leverenz, Will Harris, Austin French, Homer White, and Debbie Masden.

Thanks to Dave Shoup, Avanti Athreya, Chris Manon, Ryan Janicki, Hyejin Kim, Chris Flake, Carter Price, Charles Glover, I-kun Chen, Eleni Agathocleous, Norman Niles, Nick and Jane Long, Bob Shuttleworth, Chris Zorn, Chris Danforth, Damon Gulczynski, Matt Hoffman, Kevin Wilson, Beth McLaughlin, Russ Halper, and my other fellow graduate students for making my time in the Math Department at the University of Maryland a great and memorable experience. I am very grateful for each of your friendships. Also, I would like to thank God for blessing me with the ability to complete this dissertation, and for all of the wonderful times and friendships I've experienced along the way.

Table of Contents

List of Figures

List of Abbreviations

EM	Expectation-Maximization
CE	Cross-Entropy
MRAS	Model Reference Adaptive Search
CD	Cholesky Decomposition
s.p.d.	symmetric positive definite

Chapter 1

Introduction

1.1 Background

A mixture model is a statistical model where the probability density function is a convex sum of multiple density functions. Mixture models provide a flexible and powerful mathematical approach to modeling many natural phenomena in a wide range of fields (McLachlan and Peel, 2000). One particularly convenient attribute of mixture models is that they provide a natural framework for clustering data, where the data are assumed to originate from a mixture of probability distributions, and the cluster memberships of the data points are unknown. Mixture models are highly popular and widely applied in many fields, including biology, genetics, economics, engineering, and marketing. Mixture models also form the basis of many modern supervised and unsupervised classification methods such as neural networks or mixtures of experts.

The primary application of mixture models in this dissertation is clustering data. Mixture models are an extremely common tool in practice for clustering data to achieve many different goals. For example, in biological sequence analysis, clustering is used to group DNA sequences with similar properties. In data mining, researchers use cluster analysis to partition data items into related subsets, based on their quantifiable attributes. In social sciences, clustering may be used to recognize communities within social networks of people. These examples represent a small portion of the many applications of clustering via mixture models for real-world data.

In mixture analysis, the goal is to estimate the parameters of the underlying mixture distributions by maximizing the likelihood function of the mixture density with respect to the observed data. One of the most popular methods for obtaining the maximum likelihood estimate is the Expectation-Maximization (EM) algorithm. The EM algorithm has gained popularity in mixture analysis, primarily because of its many convenient properties. One of these properties is that it guarantees an increase in the likelihood function in every iteration (Dempster et al., 1977). Moreover, because the algorithm operates on the log-scale, the EM updates are analytically simple and numerically stable for distributions that belong to the exponential family, such as Gaussian. However, one drawback of EM is that it is a *local* optimization method only; that is, it converges to a local optimum of the likelihood function (Wu, 1983). This is a problem because with increasing data-complexity (e.g., higher dimensionality of the data and/or increasing number of clusters), the number of local optima in the mixture likelihood increases. Furthermore, the EM algorithm is a *deterministic* method; i.e., it converges to the same stationary point if initiated from the same starting value. So, depending on its starting values, there is a chance that the EM algorithm can get stuck in a sub-optimal solution, one that may be far from the global (and true) solution. The mathematical details of mixture models and the EM algorithm are given in Chapter 2 of this dissertation.

There exist many modifications of the EM algorithm that address shortcomings or limitations of the basic EM formulation. For instance, Booth and Hobert (1999) propose solutions to overcome intractable E-steps (see also Levine and Casella, 2001; Levine and Fan, 2003; Jank, 2004; Caffo et al., 2003). On the other hand, Meng and Rubin (1993) suggest ways to overcome complicated M-steps (see also Meng, 1994; Liu and Rubin, 1994). The EM algorithm is also known to converge only at a linear rate; ways to accelerate convergence have been proposed in Louis (1982), Jamshidian and Jennrich (1993), and Jamshidian and Jennrich (1997). Yet, to date, very few modifications have addressed global optimization qualities of the EM paradigm.

There have been relatively few attempts at systematically addressing the shortcomings of EM in the mixture model context. Perhaps the most common approach in practice is to simply re-run EM from multiple (e.g., randomly chosen) starting values, and then select the parameter value that provides the best solution obtained from all runs (see Biernacki et al., 2003). In addition to being computationally burdensome, especially when the parameter space is large, this approach is somewhat ad-hoc. More systematic approaches involve using *stochastic* versions of the EM algorithm such as the Monte Carlo EM (MCEM) algorithm (Wei and Tanner, 1990). Alternative approaches rely on producing ergodic Markov chains that exhaustively explore every point in the parameter space (see e.g., Diebolt and Robert, 1990; Cao and West, 1996; Celeux and Govaert, 1992). Another approach that has been proposed recently is to use methodology from the global optimization literature. In that context, Jank (2006a) proposes a Genetic Algorithm version of the MCEM algorithm to overcome local solutions in the mixture likelihood (see also Jank, 2006b; Tu et al., 2006). Along the same lines, Ueda and Nakano (1998) propose a deterministic annealing EM (DAEM) designed to overcome the local maxima problem associated with EM.

Numerous additional methods for clustering, other than simply extensions of EM, have been developed in recent years. Mangiameli et al. (1996) compare the self-organizing map (SOM) neural network with other hierarchical clustering methods. Milligan (1981) gives a computational study of many algorithms for clustering analysis, including the well-known Ward's minimum variance hierarchical procedure (Ward, Jr., 1963). However, many of these clustering procedures do not incorporate ideas from the theory of global optimization.

Two methods from the operations research literature that are designed to attain globally optimal solutions to general multi-extremal continuous optimization problems are the Cross-Entropy (CE) method (De Boer et al., 2005) and Model Reference Adaptive Search (MRAS) (Hu et al., 2007). The CE method iteratively generates candidate solutions from a parametric sampling distribution. The candidates are all scored according to an objective function, and the highest scoring candidates are used to update the parameters of the sampling distribution. These parameters are updated by taking a convex combination of the sampling parameters from the previous iteration and sample statistics of the top candidate solutions. In this way, the properties of the *best* candidates in each iteration are retained. MRAS shares similarities with CE. Like the CE method, MRAS also solves continuous optimization problems by producing candidate solutions in each iteration. However, the primary difference is that MRAS utilizes a different procedure for updating its sampling parameters, leading to a more general framework in which theoretical convergence of a particular instantiated algorithm can be rigorously proved (Hu et al., 2007). In this dissertation we propose methods that apply CE and MRAS updating principles for the global optimization of Gaussian mixture models.

1.2 Contributions of this Dissertation

Because MRAS was introduced relatively recently (in the last couple of years), there exists no work to date on applying MRAS to the global optimization of mixture models or clustering problems. Additionally, only few works have addressed applying the CE method to clustering problems. Botev and Kroese (2004) propose the CE method for Gaussian mixtures with data of small dimension, and Kroese et al. (2007) use CE in vector quantization clustering. This dissertation proposes several new algorithms that apply the ideas of CE and MRAS to maximum likelihood estimation in mixture models, and are also capable of handling high dimensional data.

One of the major difficulties for any algorithm that utilizes either CE or MRAS for the estimation of Gaussian mixture models of data with high dimension is the efficient simulation of the positive definite covariance matrices of the mixture components. This dissertation proposes several new solutions to this problem in Chapter 3. One solution is to blend the updating procedure of CE and MRAS with the principles of Expectation-Maximization updating for the covariance matrices, leading to two new algorithms, CE-EM and MRAS-EM. A second solution involves updating the Cholesky factorizations of the covariance matrices, as opposed to updating the components of the covariance matrices themselves. Using the Cholesky decomposition in the updating procedure leads to two new algorithms, CE-CD and MRAS-CD. Numerical experiments illustrate the effectiveness of the proposed algorithms in finding global optima where the classical EM fails to do so. We find that although a single run of any of the new algorithms may be slower than EM, they have the potential of producing significantly better global solutions to Gaussian mixture models. We also show that the global optimum matters in the sense that it significantly improves the clustering task (Heath et al., 2007b).

Of the many optimization algorithms that are designed to overcome locally optimal solutions, most can only offer a promise of better performance than EM in empirical studies. That is, most of the approaches stop short of guaranteeing global convergence and, similar to EM, can only guarantee convergence to a local optimum. In Chapter 4 of this dissertation we rigorously prove the convergence of the MRAS-CD algorithm to the global optimum of Gaussian mixtures. The proof gives justification that the algorithm is not merely an ad-hoc heuristic, but is a systematic approach for producing globally optimal solutions to Gaussian mixture models (Heath et al., 2007a).

Because the likelihood function of a mixture density is highly nonlinear, understanding its physical properties is difficult. In Chapter 5 of this dissertation we analyze what attributes make a global optimization problem difficult and provide evidence to why estimating Gaussian mixture models is such a difficult optimization problem. One such reason is that mixture models can have a large number of local optima that are quite inferior to the global optima. We propose and discuss metrics that quantify the difficulty of a given optimization problem, and demonstrate these metrics on several numerical examples. Furthermore, we measure how the difficulty of the optimization problem changes with varying dimensionality, number of clusters, and number of data points.

Chapter 2

Mathematical Background

2.1 Choosing the Optimal Number of Mixture Components

In practice, sometimes there is enough information available about the data in a mixture model that g, the number of mixture components, is known *a priori*. Otherwise, finding the optimal number of components in a mixture model can be a difficult problem in itself (McLachlan and Peel, 2000). In that case, it is necessary to optimize the mixture model across values of g. In this dissertation we assume that the optimal number of components g in the mixture are known, and thus focus on methods which obtain the mixture model which best fit the data for the given value of the number of components g. Methods for estimating the optimal value for g from the data are discussed in Fraley and Raftery (1998). In principle, one could combine these methods with the global optimization algorithms for mixture models that we propose in Chapter 3. The only adjustment that needs to be made is that the log-likelihood function as the optimization criterion be replaced by a suitable model-selection criterion such as the Akaike information criterion (AIC) or the Bayesian information criterion (BIC) (McLachlan and Peel, 2000).

2.2 Finite Mixture Models

We begin by presenting the mathematical framework of finite mixture models. Assume there are n observed data points, $y = \{y_1, ..., y_n\}$, in some p-dimensional space. Assume that data is known to have been derived from g distinct probability distributions, weighted according to the vector $\pi = (\pi_1, ..., \pi_g)$, where the weights are positive and sum to one. Each component of the mixture has an associated probability density $f_j(\,\cdot\,; \psi_j)$, where ψ_j represents the parameters of the j^{th} mixture component. The mixture model parameters that need to be estimated are $\theta = (\pi_j; \psi_j)_{j=1}^{g}$; that is, both the weights and the probability distribution parameters for each of the g components. We write the mixture density of the data point y_i as:

$$\tilde{f}(y_i; \theta) = \sum_{j=1}^{g} \pi_j f_j(y_i; \psi_j).$$

The typical approach to estimating the parameters θ with respect to the observed data y is via maximization of the likelihood function:

$$L(y, \theta) = \prod_{i=1}^{n} \tilde{f}(y_i; \theta),$$

which is equivalent to maximization of the log-likelihood function:

$$\begin{aligned} \ell(y, \theta) = \log L(y, \theta) &= \sum_{i=1}^{n} \log \tilde{f}(y_i; \theta) \\ &= \sum_{i=1}^{n} \log \sum_{j=1}^{g} \pi_j f_j(y_i; \psi_j). \end{aligned}$$

Maximization of the log-likelihood function in the mixture model problem is nontrivial, primarily because the log-likelihood function ℓ typically contains many local maxima, especially when the number of components g and/or the data-dimension p is large.

Consider the following example for illustration. We simulate 40 points from two univariate Gaussian distributions with means $\mu_1 = 0$ and $\mu_2 = 2$, variances $\sigma_1^2 = .001$ and $\sigma_2^2 = 1$, and each weight equal to .5. Notice that in this relatively simple example, we have 5 parameters to optimize (because the second weight is uniquely determined by the first weight). Figure 2.1 shows the log-likelihood function plotted against only one parameter-component, μ_1. All other parameters are held constant at their true values. Notice the large number of local maxima to the right of the optimal value of $\mu_1 \approx 0$. Clearly, if we start the EM algorithm at, say, 3, it could get stuck far away from the global (and true) solution. This demonstrates that a very simple situation can already cause problems with respect to global and local optima.

2.3 Model-Based Clustering

Model-based clustering is a common and natural extension of finite mixture models. The mathematical framework of model-based clustering is the same as that of finite mixture models described in Section 2.2. The mixture components are oftentimes referred to as the clusters in the model-based clustering context. After estimating the parameters of the mixture model, we can then statistically infer how the data points can be grouped into the corresponding g clusters.

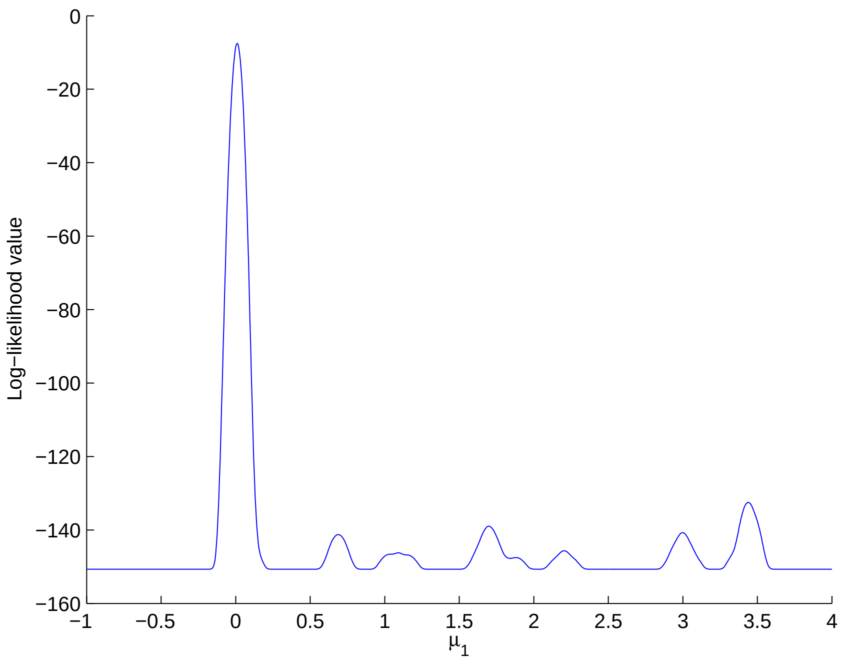

Figure 2.1: Plot of the log-likelihood function of the data set described above with parameters $\mu = (0, 2)$, $\sigma^2 = (.001, 1)$, and $\pi = (.5, .5)$, plotted against varying values of the mean component μ_1.

2.4 The Expectation-Maximization Algorithm

The Expectation-Maximization (EM) algorithm is an iterative procedure designed to produce maximum likelihood estimates in incomplete data problems (Dempster et al., 1977). We let y denote the observed (or incomplete) data, and z the unobserved (or missing) data. We refer to the collection of the observed and unobserved data (y, z) as the complete data. Let $f(y, z; \theta)$ denote the joint distribution of the complete data, where θ represents its corresponding parameter vector, which lies in the set Ω of all possible θ. EM iteratively produces estimates to the maximum likelihood estimate (MLE) of θ, denoted by θ^*, by maximizing the marginal likelihood $L(y, \theta) = \int f(y, z; \theta) dz$. Each iteration of the EM algorithm consists of an expectation and a maximization step, denoted by the E-step and M-step, respectively. Letting $\theta^{(t)}$ denote the parameter value computed in the t^{th} iteration of the algorithm, the E-step consists of computing the expectation of the complete data log-likelihood, conditional on the observed data x and the previous parameter value:

$$Q(\theta|\theta^{(t-1)}) = E\left[\log f(y, z; \theta)|y; \theta^{(t-1)}\right].$$

This conditional expectation is oftentimes referred to as the Q-function. The M-step consists of maximizing the Q-function:

$$\theta^{(t)} = \underset{\theta \in \Omega}{\operatorname{argmax}}\, Q(\theta|\theta^{(t-1)}).$$

Therefore, $Q(\theta^{(t)}|\theta^{(t-1)}) \geq Q(\theta|\theta^{(t-1)})$, and so EM guarantees an increase in the likelihood function in every iteration (Dempster et al., 1977). Given an initial estimate $\theta^{(0)}$, the EM algorithm successively alternates between the E-step and M-step to produce a sequence $\theta^{(0)}, \theta^{(1)}, \theta^{(2)}, ...$ until convergence. The stopping criterion generally used to signify convergence in the EM algorithm is when the difference or relative difference of successive log-likelihood values falls below a specified tolerance. Under mild regularity conditions (Wu, 1983), the sequence of estimates generated by EM converges to θ^*.

For the model-based clustering problem, the incomplete data are the observed data points $y = \{y_1, ..., y_n\}$. The missing, or unobserved, data are the cluster memberships of the observed data points. We write the missing data as $z = \{z_1, ..., z_n\}$, where z_i is a g-dimensional $0-1$ vector such that $z_{ij} = 1$ if the observed data point y_i belongs to cluster j, and $z_{ij} = 0$ otherwise. In other words, $z_{ij} = 1$ signifies that y_i was generated from the probability density $f(\,\cdot\,;\psi_j)$. We can now write the complete data log-likelihood of $\theta = (\pi;\psi)$ as

$$\log L(y,\theta) = \sum_{i=1}^{n}\sum_{j=1}^{g} z_{ij}\left\{\log \pi_j + \log f(y_i;\psi_j)\right\}.$$

The Gaussian mixture model allows significant simplifications for the EM updates. In fact, in the Gaussian case both the E-step and M-step can be written in closed form (Jank, 2006b). In the E-step of each iteration, we compute the conditional expectation of the components of z with respect to the observed data x and the parameters of the previous iteration $\theta^{(t-1)} = \left(\pi^{(t-1)};\mu^{(t-1)},\Sigma^{(t-1)}\right)$ by

$$\begin{aligned}\tau_{ij}^{(t-1)} &= E\left(z_{ij}|y_i;\theta^{(t-1)}\right)\\ &= \frac{\pi_j^{(t-1)}\phi(y_i;\mu_j^{(t-1)},\Sigma_j^{(t-1)})}{\sum_{c=1}^{g}\pi_c^{(t-1)}\phi(y_i;\mu_c^{(t-1)},\Sigma_c^{(t-1)})} \qquad (2.1)\end{aligned}$$

for all $i = 1,...,n$ and $j = 1,...,g$, where $\phi(\,\cdot\,;\mu,\Sigma)$ is the Gaussian density function with mean μ and covariance matrix Σ. Next, we compute the sufficient statistics:

$$T_{j1}^{(t)} = \sum_{i=1}^{n}\tau_{ij}^{(t-1)}, \qquad (2.2)$$

$$T_{j2}^{(t)} = \sum_{i=1}^{n}\tau_{ij}^{(t-1)}y_i, \qquad (2.3)$$

$$T_{j3}^{(t)} = \sum_{i=1}^{n}\tau_{ij}^{(t-1)}y_i y_i^T. \qquad (2.4)$$

The M-step consists of updating the Gaussian parameters by means of the above sufficient statistics:

$$\pi_j^{(t)} = \frac{T_{j1}^{(t)}}{n}, \qquad (2.5)$$

$$\mu_j^{(t)} = \frac{T_{j2}^{(t)}}{T_{j1}^{(t)}}, \qquad (2.6)$$

$$\Sigma_j^{(t)} = \frac{T_{j3}^{(t)} - T_{j1}^{(t)^{-1}}T_{j2}^{(t)}T_{j2}^{(t)^T}}{T_{j1}^{(t)}}. \qquad (2.7)$$

EM is a deterministic algorithm; that is, the solution generated by EM is determined solely by the starting value $\theta^{(0)}$. Subsequent runs from the same starting value will lead to the same solution. Also, EM is a locally converging algorithm, and so depending on its starting value, EM may not produce the global optimizer of the likelihood function. We demonstrate this on the example from Section 2.2, choosing three different starting values. In particular, we let $\mu_2 = 2$, $\sigma^2 = (.001, 1)$, and $\pi = (.5, .5)$ for each initialization, along with three different values of μ_1: 0, 2, and 3. Figure 2.2 shows the iteration paths of EM for each of the three starting values. In this example we choose the stopping criterion to be $|\zeta_k - \zeta_{k-5}| \leq 10^{-5}$, where ζ_k is the log-likelihood value obtained in iteration k. As might be expected from the shape of Figure 2.1, only the run with $\mu_1 = 0$ produces the globally optimal solution, while the other two runs converge to sub-optimal solutions.

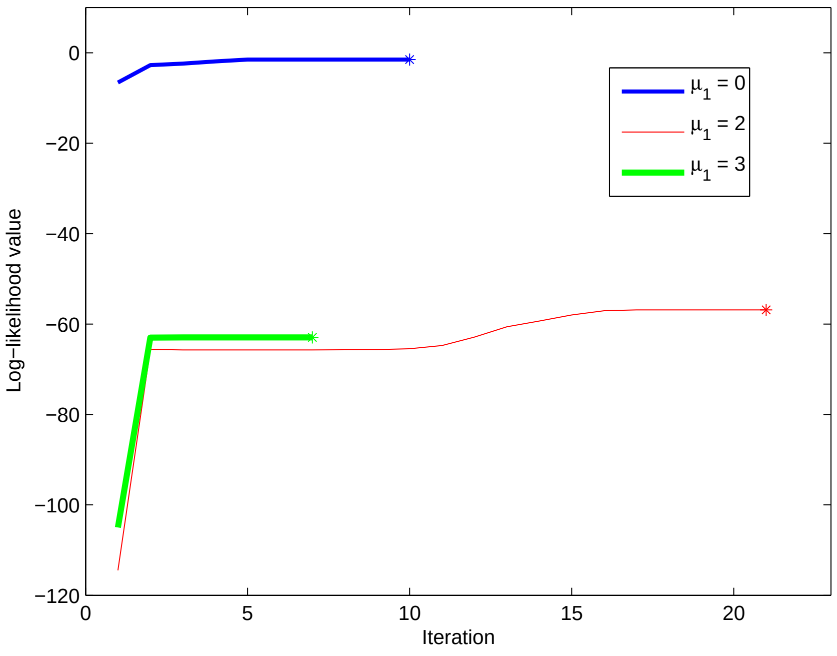

Figure 2.2: Plot of the iteration paths for the log-likelihood of 3 different runs of EM on the data set described in Section 2.2, with each run initialized with a different value of μ_1. The $*$ denotes the log-likelihood value reached at convergence.

Chapter 3

New Global Optimization Algorithms for Model-Based Clustering

3.1 Motivation

Two methods from the operations research literature that are designed to attain globally optimal solutions to general multi-extremal continuous optimization problems are the Cross-Entropy (CE) method (De Boer et al., 2005) and Model Reference Adaptive Search (MRAS) (Hu et al., 2007). Both the CE method and MRAS iteratively generate candidate solutions from a parametric sampling distribution. The primary difference between the two are their different procedures for updating their corresponding sampling parameters. In this chapter we set out to apply these powerful global optimization methods to Gaussian mixture models and the model-based clustering setting. The main purpose of doing so is because the classical method for producing solutions to Gaussian mixture models in practice is the locally converging EM algorithm. While the EM algorithm is capable of relatively quick convergence, it only guarantees convergence to a local optimum of the likelihood function (Wu, 1983). The likelihood of the mixture density may contain many such local optima. We propose four new algorithms designed to overcome locally optimal solutions of mixture models, two of which combine the convenient properties of the EM paradigm with the ideas underlying global optimization.

The main contribution of this chapter is the development of global and efficient methods that utilize ideas of CE and MRAS to find globally optimal solutions for model-based clustering problems. Our primary goal is to achieve better solutions to clustering problems when the classical EM algorithm only attains locally optimal solutions. In that context, one complicating factor is maintaining the positive definiteness of the mixture-model covariance matrices, especially for high-dimensional data. We describe a previously-proposed method by Botev and Kroese (2004) and demonstrate implementation issues that arise with data of dimension greater than two. The primary problem is that simulating the covariance matrices directly becomes highly constrained as the dimension increases. We propose alternate updating procedures that ensure the positive definiteness in an efficient way. One of our solutions applies principles of EM for the updating scheme of the covariance matrices to produce the CE-EM and MRAS-EM algorithms. Additionally, we exploit the work of unconstrained parameterization of covariance matrices (Pinheiro and Bates, 1996) to produce the CE-CD and MRAS-CD algorithms based on the Cholesky decomposition. Chapter 4 of this dissertation focuses on proving theoretical convergence of MRAS-CD to the global optimum of Gaussian mixture models. However, proving theoretical convergence of the other three algorithms proposed in this chapter is still an open problem.

We apply our methods to several simulated and real data sets and compare their performance to the classical EM algorithm. We find that although a single run of the global optimization algorithms may be slower than EM, all have the potential of producing significantly better solutions to the model-based clustering problem. We also show that the global optimum "matters", in that it leads to improved decision making, and, particularly in the clustering context, to significantly improved clustering decisions.

The rest of the chapter begins with explaining the CE method and MRAS in Section 3.2, which provides the framework for our discussion of the proposed CE-EM, CE-CD, MRAS-EM, and MRAS-CD algorithms. In Section 3.3 we carry out numerical experiments to investigate how these new global optimization approaches perform in the model-based clustering problem with respect to the classical EM algorithm. The examples include simulated data sets and one real-world survey data set. We conclude and discuss future work in Section 3.4.

3.2 Global Optimization Methods

In the following we discuss two powerful global optimization methods from the operations research literature. We describe the methods and also the challenges that arise when applying them to the model-based clustering context. We then propose several new algorithms to overcome these challenges.

3.2.1 The Cross-Entropy Method

The Cross-Entropy (CE) method is a global optimization method that relies on iteratively generating candidate solutions from a sampling distribution, scoring the candidates according to the objective function, and updating the sampling distribution with sample statistics of the highest-scoring candidates. The CE method has been used in a variety of discrete optimization settings such as rare event simulation (Rubinstein, 1997) and combinatorial optimization problems (Rubinstein, 1999), as well as continuous optimization problems (Kroese et al., 2006). However, applying CE to model-based clustering problems is a relatively new idea (Botev and Kroese, 2004).

For the Gaussian model-based clustering problem described in Chapter 2, we are trying to find the maximum likelihood estimate for the mixture density of n p-dimensional data points across g clusters. To that end, we need to estimate the unknown cluster parameters: the mean vector μ_j, covariance matrix Σ_j, and weight π_j for each cluster $j = 1, ..., g$. Therefore, when we apply the CE method to the clustering setting, we need to generate g mean vectors, g covariance matrices, and g weights for each candidate. Generating valid covariance matrices randomly is non-trivial, which we will discuss in detail in Section 3.2.2. Note that since the covariance matrix is symmetric, it is sufficient to work with a $p(p+1)/2$-dimensional vector to construct a $p \times p$ covariance matrix.

As pointed out above, it is necessary to generate the following cluster parameters for each candidate: $g \cdot p$ cluster means, $g \cdot p(p+1)/2$ components used for the construction of the g covariance matrices, and g weights, yielding a total of $g(p+2)(p+1)/2$ cluster parameters. By convention, we let the candidate solution X be a vector comprised of the $g(p+2)(p+1)/2$ cluster parameters. Our goal is to generate the optimal vector X^* that contains the cluster parameters of the maximum likelihood estimate $\theta^* = (\mu_j^*, \Sigma_j^*, \pi_j^*)_{j=1}^g$ of the mixture density.

In each iteration we generate N candidate vectors $X_1, ..., X_N$ according to a certain sampling distribution. The CE literature (Kroese et al., 2006) for continuous optimization suggests generating the components of each candidate independently, using the Gaussian, double-exponential, or beta distributions, for example. We choose the sampling distribution to be Gaussian for the simplicity of its parameter updates. Therefore, X_i is drawn from $\mathrm{N}(a, b^2 I)$, where a is the $\frac{g(p+2)(p+1)}{2}$-dimensional mean vector, $b^2 I$ is the corresponding $\frac{g(p+2)(p+1)}{2} \times \frac{g(p+2)(p+1)}{2}$ covariance matrix. We note that all off-diagonal components of $b^2 I$ are zero, and so the components of X_i are generated independently. As we will discuss later, it is also necessary to generate some of the components of X_i from a truncated Gaussian distribution. The next step is to compute the log-likelihood values of the data with respect to each set of candidate cluster parameters. In each iteration, a fixed number of candidates with the highest corresponding log-likelihood values, referred to as the elite candidates, are used to update the sampling distribution parameters (a, b). The new sampling parameters are updated in a *smooth* manner by taking a convex combination of the previous sampling parameters with the sample mean and sample standard deviation of the elite candidate vectors. In the following, we describe each of these steps in detail.

The CE algorithm for mixture models can be seen in Figure 3.1. In the algorithm, the number of candidate solutions generated at each iteration is fixed at N and the number of elite samples taken at each iteration is fixed at N^{elite}. The smoothing parameters α and β used in the updating step are also fixed. Note that setting $\alpha = 1$ updates the sampling mean with the value of the mean of the elite candidates in that iteration. Doing so may lead to premature convergence of the algorithm, resulting in a local, and poor, solution. Using a value of α between .5

and .9 results in better performance, as the updated sampling mean incorporates the previous sampling mean. Similarly, choosing a value of β close to 1 will accelerate convergence, and so a value chosen between .3 and .5 seems to perform better, as noted by emprirical evidence. The algorithm returns the candidate solution that produces the highest log-likelihood score among all candidates in all iterations.

```
  **Data**: Data points $y_1, y_2, ..., y_n$
  **Result**: Return highest-scoring estimate for $X^*$.
1 Initialize $a_0$ and $b_0^2$.
2 $k \Leftarrow 1$.
3 repeat
4     Generate $N$ i.i.d. candidate vectors $X_1, ..., X_N$ from the sampling
      distribution $\mathrm{N}(a_{k-1}, b_{k-1}^2 I)$.
5     Compute the log-likelihoods $\ell(y, X_i), ..., \ell(y, X_N)$.
6     For the top-scoring $N^{elite}$ candidate vectors, let $\tilde{a}_k$ be the vector of
      their sample means, and let $\tilde{b}_k^2$ be the vector of their sample variances.
7     Update $a_k$ and $b_k$ in a smooth way according to:
          $a_k = \alpha\, \tilde{a}_k + (1-\alpha) a_{k-1},$
          $b_k = \beta\, \tilde{b}_k + (1-\beta) b_{k-1}.$
8     $k \Leftarrow k + 1$.
9 until Stopping criterion is met.
```

Figure 3.1: CE Algorithm for Mixture Models

The main idea of this algorithm is that the sequence $a_0, a_1, ...$ will converge to the optimal vector X^* representing the MLE $\theta^* = (\mu_j^*, \Sigma_j^*, \pi_j^*)_{j=1}^g$ as the sequence of the variance vectors $b_0^2, b_1^2, ...$ converges to zero. The stopping criterion we use is to stop the algorithm when the best log-likelihood value over k iterations does not increase by more than a specified tolerance. However, occasionally one or more components of the variances of the parameters prematurely converges to zero, perhaps at a local maximum. One way to deal with this is by "injecting" extra variance into the sampling parameters b_k^2 when the maximum of the diagonal components in $b_k^2 I$ is less than a fixed threshold (Rubinstein and Kroese, 2004). By increasing the variance of the sampling parameters, we expand the sampling region to avoid getting trapped in a locally sub-optimal solution. We provide a list of both the model parameters and the CE parameters in Table 3.1.

Table 3.1: List of the model and CE parameters.

Mixture Model parameters	CE parameters
n = number of data points	N = number of candidates
y_i = i^{th} data point	X_i = candidate vector
p = dimension of data	N^{elite} = number of elite candidates
g = number of mixture components	a_k = Gaussian sampling mean vector
π_j = weight of j^{th} mixture component	$b_k^2 I$ = Gaussian sampling covariance matrix
ψ_j = probability distribution parameters of j^{th} component	X^* = candidate vector representing the global optimum
$f_j(\,\cdot\,;\psi_j)$ = probability density of j^{th} component	$\tilde{a}_k$ = mean of elite candidates in k^{th} iteration
θ = model parameters to estimate	$\tilde{b}_k^2$ = variance vector of elite candidates in k^{th} iteration
θ^* = model parameters that represent the global optimum	α = smoothing parameter for the sampling means
$\ell(y,\theta)$ = log-likelihood function	β = smoothing parameter for the sampling standard deviations
μ_j = Gaussian mixture mean vector	
Σ_j = Gaussian mixture covariance matrix	

3.2.1.1 Original CE Mixture Model Algorithm

We now discuss a potential way of generating candidates in the CE mixture-model algorithm (see e.g., Botev and Kroese, 2004). The unknown parameters of the Gaussian mixture model that we are estimating are the cluster means, the cluster covariance matrices, and the cluster weights. Let us take the case where the data is 2-dimensional, such that each cluster j has two components of the mean vector, $\mu_{j,1}$ and $\mu_{j,2}$, a 2×2 covariance matrix Σ_j, and one cluster weight π_j. We can construct the covariance matrix for each cluster by simulating the variances of each component, $\sigma^2_{j,1}$ and $\sigma^2_{j,2}$, and their corresponding correlation coefficient, ρ_j, and then populating the covariance matrix as follows:

$$\Sigma_j = \begin{pmatrix} \sigma^2_{j,1} & \rho_j \sigma_{j,1} \sigma_{j,2} \\ \rho_j \sigma_{j,1} \sigma_{j,2} & \sigma^2_{j,2} \end{pmatrix}. \tag{3.1}$$

So, in the 2-d case, one needs to simulate 6 random variates for each cluster, resulting in $6 \times g$ random variates for each candidate.

Note that some of the model parameters must be simulated from constrained regions. Specifically, the variances must all be positive, the correlation coefficients must be between -1 and 1, and the weights must be positive and sum to one. One way to deal with the weight-constraints is via simulating only $g - 1$ weights; in this case one must ensure that the sum of the simulated $g-1$ weights is less than one. We choose a different approach. In order to reduce the constraints on the parameters generated for each candidate, we choose to instead simulate g positive weights for each candidate and then normalize the weights.

3.2.2 Challenges of the CE Mixture Model Algorithm

The cluster means and weights can be simulated in a very straightforward manner in the CE mixture model algorithm. However, generating random covariance matrices can be tricky, because covariance matrices must be symmetric and positive semi-definite. Ensuring the positive semi-definite constraint becomes increasingly difficult as the data-dimension increases. In the CE mixture model algorithm, when the dimension is greater than two, the method of populating a covariance matrix by simulating the variance components and the correlation coefficients becomes problematic. This issue is not addressed in the original paper introducing the CE mixture model algorithm of Botev and Kroese (2004). We propose several solutions to this problem.

For practical purposes, we focus on methods that produce symmetric positive definite covariance matrices, since a covariance matrix is positive semi-definite only in the degenerate case. Ensuring the positive definite property when generating these matrices is a difficult numerical problem, as Pinheiro and Bates (1996) discuss. To investigate this problem, we rely on the following theorem, where A_i is the $i \times i$ submatrix of A consisting of the "intersection" of the first i rows and columns of A (Johnson, 1970):

Theorem 3.1 *A symmetric $n \times n$ matrix A is symmetric positive definite (s.p.d.) if and only if* $\det A_i > 0$ *for* $i = 1, ..., n$.

Consider again the 2-dimensional case. This case is trivial, because the determinant of the covariance matrix Σ_j in Equation (2.7) is given by $\sigma_{j,1}^2 \sigma_{j,2}^2 (1 - \rho_j^2) > 0$, for all $|\rho_j| < 1$. In other words, in the 2-d case, we can construct an s.p.d. covariance matrix simply by simulating positive variances and correlation coefficients in the interval $[-1, 1]$. However, when the number of dimensions is more than two, using the same method of populating the covariance matrix from simulated variances and correlation coefficients no longer guarantees positive-definiteness. Therefore, it is necessary to place additional constraints on the correlation coefficients. Consider the example of a general 3-dimensional covariance matrix:

$$\begin{pmatrix} \sigma_1^2 & \rho_1\sigma_1\sigma_2 & \rho_2\sigma_1\sigma_3 \\ \rho_1\sigma_1\sigma_2 & \sigma_2^2 & \rho_3\sigma_2\sigma_3 \\ \rho_2\sigma_1\sigma_3 & \rho_3\sigma_2\sigma_3 & \sigma_3^2 \end{pmatrix}. \tag{3.2}$$

The matrix has determinant $\sigma_1^2\sigma_2^2\sigma_3^2 \cdot (1 + 2\rho_1\rho_2\rho_3 - \rho_1^2 - \rho_2^2 - \rho_3^2)$. One can see that simulating the three correlation coefficients on the constrained region $[-1, 1]$ will no longer guarantee a covariance matrix with a positive determinant, e.g., by choosing $\rho_1 = 1, \rho_2 = 0, \rho_3 = -1$. As the number of dimensions of the data increases, there are an increased number of constraints on a feasible set of correlation coefficients. Generating a positive definite matrix this way has shown to be computationally inefficient because of the high number of constraints (Pinheiro and Bates, 1996). To illustrate this, we conduct a numerical experiment where we generate random positive variance components and uniform random $(0, 1)$ correlation coefficients to construct covariance matrices of varying dimensions. For each dimension, we generate 100,000 symmetric matrices in this manner and then test them to see if they are positive definite (see Table 3.2). Naturally, matrices of 2 dimensions constructed this way will always be positive definite. But, as Table 3.2 indicates, this method of generating covariance matrices is not efficient for high dimensional matrices. In fact, in our experiment not a single 10-dimensional matrix out of the 100,000 generated is positive definite.

Table 3.2: The results from the experiment where 100,000 symmetric matrices of varying dimensions are generated in the same manner as in the original CE mixture model algorithm.

Dimension	Number s.p.d.	% s.p.d.
2	100,000	100
3	80,107	80.1
4	42,588	42.6
5	13,052	13.1
6	2,081	2.08
7	159	.159
8	5	.005
9	1	.001
10	0	0

3.2.3 Two New CE Mixture Model Algorithms

In this section we introduce two new CE mixture model algorithms that modify the method of generating random covariance matrices and hence overcome the numerical challenges mentioned in the previous section.

3.2.3.1 CE-EM algorithm

The problem of non-s.p.d. covariance matrices does not present itself in the EM algorithm, because the construction of the covariance matrices (2.7) in each EM iteration based on the sufficient statistics (2.2)-(2.4) guarantees both symmetry and positive-definiteness. Therefore, one potential solution to the CE mixture model algorithm is to update the covariance matrices at each iteration using the same methodology as in the EM algorithm. In this method, which we refer to as the CE-EM algorithm, the means and weights are updated via the CE algorithm for each candidate during an iteration, while we generate new covariance matrices via

EM updating.

The CE-EM algorithm has the same structure as the CE algorithm, except that the candidate vector X consists of only the cluster means and weights. Therefore, the sampling parameters a and b now have $g \cdot (p+1)$ components. In iteration k, we produce N candidates for the cluster means and weights, and we score each of the candidates along with the same cluster covariance matrices, $\Sigma^{(k-1)}$, produced in the previous iteration. The sampling parameters (a, b) are updated as in step 6 in the CE algorithm. Then, we use the cluster means and weights from the updated sampling mean a_k along with the covariance matrices from the previous iteration to compute the sufficient statistics (2.2)-(2.4) used in the EM algorithm, and then update the covariance matrices $\Sigma^{(t)}$ by (2.7). We provide a detailed description of the CE-EM algorithm in Figure 3.2.

3.2.3.2 CE-CD Algorithm

In addition to the CE-EM algorithm, we propose an alternative method to simulate positive definite covariance matrices in the CE mixture model algorithm. Pinheiro and Bates (1996) propose five parameterizations of covariance matrices in which the parameterizations ensure positive-definiteness. We adopt two of these parameterizations for our efforts, and both rely on the following theorem (Thisted, 1988) regarding the Cholesky decomposition of a symmetric positive definite matrix:

Theorem 3.2 *A real, symmetric matrix A is s.p.d. if and only if it has a Cholesky Decomposition such that $A = U^T U$, where U is a real-valued upper triangular matrix.*

```
   Data: Data points $y_1, y_2, ..., y_n$
   Result: Return highest-scoring estimate for $X^*$.
1  Initialize $a_0$, $b_0^2$, and $\Sigma^{(0)}$.
2  $k \Leftarrow 1$.
3  repeat
4      Generate $N$ i.i.d. candidate vectors $X_1, ..., X_N$ from the sampling
       distribution $\mathrm{N}(a_{k-1}, b_{k-1}^2 I)$.
5      Compute the log-likelihoods $\ell(y, X_i, \Sigma^{(k-1)}), ..., \ell(y, X_N, \Sigma^{(k-1)})$.
6      For the top-scoring $N^{elite}$ candidate vectors, let $\tilde{a}_k$ be the vector of
       their sample means, and let $\tilde{b}_k^2$ be the vector of their sample variances.
7      Update $a_k$ and $b_k$ in a smooth way according to:
           $a_k = \alpha\, \tilde{a}_k + (1 - \alpha) a_{k-1},$
           $b_k = \beta\, \tilde{b}_k + (1 - \beta) b_{k-1}.$
8      Compute sufficient statistics (2.2)-(2.4) using cluster means and
       weights in $a_k$ and $\Sigma^{(k-1)}$.
9      Update $\Sigma^{(k)}$ according to (2.7).
10     $k \Leftarrow k + 1$.
11 until Stopping criterion is met.
```

Figure 3.2: CE-EM Algorithm

Because covariance matrices are s.p.d., each covariance matrix has a corresponding Cholesky factorization U. Therefore, one possible way to stochastically generate covariance matrices in the CE mixture model is to generate the components of the U matrix from the Cholesky decomposition instead of the components of the covariance matrix Σ itself. Note that only the $\frac{p(p+1)}{2}$ upper right-hand components of U must be generated for each $p \times p$ covariance matrix (all other components are necessarily zero). Then, the covariance matrix can be constructed from the simulated Cholesky factors, ensuring that the covariance matrix is s.p.d. We will refer to this version of the CE method that updates the covariance matrices via the Cholesky decomposition as the CE-CD algorithm.

One potential problem with this method is that the Cholesky factor for a symmetric positive definite matrix is not unique. For a Cholesky factor U of Σ, we can multiply any subset of rows of U by -1 and obtain a different Cholesky factor of the same Σ. Thus there is not a unique optimal X^* in the CE-CD algorithm. This can present a problem if we generate candidate vectors X_i and X_j of the components of U and $-U$ in the CE-CD algorithm. Although the two candidate vectors represent the same covariance matrix, the benefit of using them to update the sampling parameters would offset. Different factorizations of Σ can steer the algorithm in opposite directions, because if one candidate vector contains U and another contains $-U$, their mean is zero, making convergence to a single Cholesky factor of Σ slow.

Pinheiro and Bates (1996) point out that if the diagonal elements of the Cholesky factor U are required to be positive, then the Cholesky factor U is unique. Thus, by restricting the diagonal elements of U to be positive, we can circumvent the uniqueness problem of the Cholesky factorization mentioned above. So, in the CE-CD algorithm, we choose to sample X from a truncated Gaussian distribution, where we restrict the components corresponding to the diagonal elements of U to be positive.

One drawback to implementing this method is the computation time to convergence. In comparison to the alternative method of generating covariance matrices at each iteration via the CE-EM algorithm, the computation time is increased, due to the extra burden of simulating $\frac{p(p+1)}{2}$ components to be used for the construction of the covariance matrices *for each candidate* in each iteration. In other words, only one covariance matrix is generated in each iteration in the CE-EM algorithm, while N, or the number of candidates, covariance matrices are generated in each iteration of the CE-CD algorithm.

3.2.4 Model Reference Adaptive Search

Model Reference Adaptive Search (MRAS) is a global optimization tool similar in nature to the CE method in that it generates candidate solutions from a sampling distribution in each iteration. It differs from CE in the specification of the sampling distribution and the method it uses to update the sampling parameters, which leads to a provably globally convergent algorithm (Hu et al., 2007). The sampling distribution we use is multivariate Gaussian, and thus the components generated in each iteration will be inherently correlated. We refer to the MRAS sampling distribution as $\tilde{g}(\,\cdot\,;\xi,\Omega)$, where ξ and Ω are the mean and covariance matrix of the MRAS sampling distribution, respectively. These parameters are updated iteratively using a sequence of intermediate reference distributions.

The basic methodology of MRAS can be described as follows. In the k^{th} iteration, we generate N_k candidate solutions, $X_1, X_2, ..., X_{N_k}$, according to the sampling distribution $\tilde{g}(\,\cdot\,;\xi^{(k)},\Omega^{(k)})$. After sampling the candidates, we score them according to the objective function, i.e., we compute the objective function value $H(X_i)$ for each candidate X_i. We then obtain an *elite* pool of candidates by selecting the top ρ-percentile scoring candidates. The value of ρ changes over the course of the algorithm to ensure that the current iteration's candidates improve upon the candidates in the previous iteration. Let the lowest objective function score among the elite candidates in any iteration k be denoted as γ_k. We introduce a parameter ϵ, a very small positive number, to ensure that the increment in the $\{\gamma_k\}$ sequence is strictly bounded below. If $\gamma_k < \gamma_{k-1} + \epsilon$, increase ρ until $\gamma_k \geq \gamma_{k-1} + \epsilon$, effectively reducing the number of elite candidates. If, however, no such percentile ρ exists, then the number of candidates is increased in the next iteration by a factor of α (where $\alpha > 1$), such that $N_{k+1} = \alpha N_k$.

The MRAS mixture model algorithm can be seen in Figure 3.3. In the description, note that MRAS utilizes $S : \Re \rightarrow \Re^+$, a strictly increasing function, to account for cases where the objective function value $H(X)$ is negative for a given X. Additionally, the parameter λ is a small constant which assigns a probability to sample from the initial sampling distribution $g(\,\cdot\,; \xi^{(0)}, \Omega^{(0)})$ in any subsequent iteration. $I_{\{\cdot\}}$ denotes the indicator function such that:

$$I_{\{A\}} := \begin{cases} 1, & \text{if event A holds,} \\ 0, & \text{otherwise.} \end{cases}$$

The main idea of MRAS is analogous to that of CE; i.e., the sequence of means $\xi^{(0)}, \xi^{(1)}, ...$ will converge to the optimal vector X^*, as the sequence of the sampling covariance matrices $\Omega^{(0)}, \Omega^{(1)}, ...$ converges to the zero matrix. We use the same stopping criterion for the MRAS mixture model algorithm as in CE: the algorithm stops when the increase of the best log-likelihood value over k iterations falls below a specified tolerance. Table 3.3 provides a list of the model parameters and the MRAS parameters.

```
   Data: Data points $y_1, y_2, ..., y_n$
   Result: Return highest-scoring estimate for $X^*$.
1  Initialize $\xi^{(0)}$ and $\Omega^{(0)}$.
2  $k \Leftarrow 0$.
3  repeat
4      Generate $N_k$ i.i.d. candidate vectors $X_1, ..., X_{N_k}$ from the sampling
       distribution $\tilde{g}(\,\cdot\,; \xi^{(k)}, \Omega^{(k)}) := (1-\lambda)g(\,\cdot\,; \xi^{(k)}, \Omega^{(k)}) + \lambda g(\,\cdot\,; \xi^{(0)}, \Omega^{(0)})$.
5      Compute the log-likelihoods values $\ell(y, X_1), \ell(y, X_2), ..., \ell(y, X_{N_k})$.
6      Select the elite candidates by taking the top scoring $\rho_{k-1}$-percentile
       candidate vectors, and define $\tilde{\gamma}_k(\rho_k)$ as the $\rho_k$-percentile log-likelihood
       score obtained of all candidates in iteration $k$.
7      if $k = 0$ or $\tilde{\gamma}_k(\rho_k) \geq \gamma_k + \frac{\epsilon}{2}$ then
8          $\gamma_{k+1} \Leftarrow \tilde{\gamma}_k(\rho_k), \rho_{k+1} \Leftarrow \rho_k$, and $N_{k+1} \Leftarrow N_k$.
9      else
10         find the largest $\tilde{\rho} \in (\rho_k, 100)$ such that $\tilde{\gamma}_k(\tilde{\rho}) \geq \gamma_k + \frac{\epsilon}{2}$.
11         if such a $\tilde{\rho}$ exists then
12             $\gamma_{k+1} \Leftarrow \tilde{\gamma}_k(\tilde{\rho})$, $\rho_{k+1} \Leftarrow \tilde{\rho}$, and $N_{k+1} \Leftarrow N_k$,
13         else
14             $\gamma_{k+1} \Leftarrow \gamma_k$, $\rho_{k+1} \Leftarrow \rho_k$, and $N_{k+1} \Leftarrow \alpha N_k$.
15         end
16     end
17     Update the sampling parameters according to:

       $$\xi^{(k+1)} = \frac{\sum_{i=1}^{N_k} S(\ell(y,X_i))^k / \tilde{g}(X_i, \xi^{(k)}, \Omega^{(k)}) I_{\{\ell(y,X_i) \geq \gamma_{k+1}\}} X_i}{\sum_{i=1}^{N_k} S(\ell(y,X_i))^k / \tilde{g}(X_i, \xi^{(k)}, \Omega^{(k)}) I_{\{\ell(y,X_i) \geq \gamma_{k+1}\}}},$$
       $$\Omega^{(k+1)} = \frac{\sum_{i=1}^{N_k} \frac{S(\ell(y,X_i))^k}{\tilde{g}(X_i, \xi^{(k)}, \Omega^{(k)})} I_{\{\ell(y,X_i) \geq \gamma_{k+1}\}} (X_i - \xi^{(k+1)})(X_i - \xi^{(k+1)})^T}{\sum_{i=1}^{N_k} S(\ell(y,X_i))^k / \tilde{g}(X_i, \xi^{(k)}, \Omega^{(k)}) I_{\{\ell(y,X_i) \geq \gamma_{k+1}\}}}.$$

18     $k \Leftarrow k + 1$.
19 until Stopping criterion is met.
```

Figure 3.3: MRAS mixture model algorithm

3.2.5 Two New MRAS mixture model algorithms

Analogous to the CE mixture model algorithms, we introduce two new MRAS mixture model algorithms that overcome the difficulties of generating random covariance matrices, namely MRAS-EM and MRAS-CD.

Table 3.3: List of the model and MRAS parameters.

Mixture Model parameters	MRAS parameters
n = number of data points	N_k = number of candidates in k^{th} iteration
y_i = i^{th} data point	X_i = candidate vector
p = dimension of data	$\xi^{(k)}$ = Gaussian sampling mean
g = number of mixture components	$\Omega^{(k)}$ = Gaussian sampling covariance matrix
π_j = weight of j^{th} mixture component	$\tilde{g}(\ \cdot\ ;\xi^{(k)},\Omega^{(k)})$ = Gaussian sampling density
ψ_j = probability distribution parameters of j^{th} component	γ_k = lowest objective score of elite candidates in k^{th} iteration
$f_j(\ \cdot\ ;\psi_j)$ = probability density of j^{th} component	ρ = elite candidate percentile
θ = model parameters to estimate	α = multiplicative parameter
θ^* = model parameters that represent the global optimum	X^* = candidate vector representing the global optimum
$\ell(y,\theta)$ = log-likelihood function	λ = sampling weight
μ_j = Gaussian mixture mean vector	$S : \Re \to \Re^+$ = strictly increasing function
Σ_j = Gaussian mixture covariance matrix	ϵ = lower bound on the increase of each γ_k

3.2.5.1 MRAS-EM Algorithm

The methodology of the MRAS-EM algorithm is parallel to that of the CE-EM algorithm. Because we are dealing with the same issue of how to stochastically construct and update covariance matrices in each iteration while maintaining the symmetric positive-definite property of these matrices, the same algorithmic scheme can be used as discussed in Section 3.2.3.1 on the CE-EM algorithm. Therefore, MRAS updating is used for the cluster means and weights, while the cluster covariance matrices are updated via the EM algorithm. Note that in each iteration the cluster means and weights are updated first and are then used to update the covariance matrices.

3.2.5.2 MRAS-CD Algorithm

With the MRAS-CD algorithm, we use the same ideas as in the CE-CD algorithm as discussed in Section 3.2.3.2. In other words, the covariance matrices are decomposed into its the components of the Cholesky decomposition, which are then simulated for each candidate in each iteration, and then used for the construction of the cluster covariance matrices. It is the Cholesky decomposition components, along with the cluster means and weights, that constitute the model parameters that are simulated and used for updating purposes in each iteration of the MRAS-CD algorithm. To ensure uniqueness of the Cholesky decomposition, we sample the diagonal components of the Cholesky factorizations from the interval $[0, \infty]$. Chapter 4 provides a proof of convergence of MRAS-CD to the global optimum for Gaussian mixtures.

3.3 Numerical Experiments

In the following numerical experiments, we demonstrate the performance of the proposed algorithms in comparison with the original EM algorithm. To that end, we design three different experiments of increasing complexity. All experiments are performed in Matlab and are run on a 2.80 GHz Intel with 1 GB RAM.

3.3.1 Preventing Degenerate Clusters

Maximizing the log-likelihood function in the Gaussian mixture model can lead to unbounded solutions, if the parameter space is not properly constrained.

In fact, we can make the log-likelihood value arbitrarily large by letting one of the component means be equal to a single data point, and then letting the generalized variance, or determinant of the covariance matrix, of that component be arbitrarily small. Such a solution is referred to as a degenerate, or spurious, solution. In order to prevent degenerate solutions in practice, it is necessary to constrain the parameter space in such a way as to avoid exceedingly small variance components in the univariate case, or exceedingly small generalized variances in the multivariate case.

One constraint that achieves this goal is to limit the relative size of the generalized variances of the mixture components (McLachlan and Peel, 2000) and it is given by:

$$\min_{i,j} \frac{|\Sigma_i|}{|\Sigma_j|} \geq c > 0,$$

where $|\Sigma|$ denotes the determinant of the matrix Σ. To avoid degenerate solutions, we will use the following constraint instead:

$$\min_{j} |\Sigma_j| \geq c > 0. \tag{3.3}$$

In each of these constraints, determining the appropriate value of c is difficult when no prior information on the problem structure is known. For our numerical experiments, we use a value of $c = .01$. If any algorithm generates a covariance matrix that violates the constraint given by (3.3), we discard it and re-generate a new one.

3.3.2 Initial Parameters

For the EM algorithm we use uniform starting values over the solution space. That is, we initialize the means uniformly over the range of the data, we initialize the variances uniformly between 0 and the sample variance of the data, and we initialize the weights uniformly between 0 and 1. Then we normalize the weights so that they sum to one. The stopping criterion for the EM algorithm is set to $|\zeta_k - \zeta_{k-1}| \leq 10^{-5}$, where ζ_k is the log-likelihood value obtained in iteration k.

One of the benefits of the CE method (and MRAS, for that matter) is that its performance is virtually independent of its starting values for many practical purposes (De Boer et al., 2005). We initialize the parameters a_0 and $\xi^{(0)}$ of the CE- and MRAS-based algorithms as follows: we set the means equal to the mean of the data, we set the covariance matrices equal to diagonal matrices with the sample variances of the data along the diagonals, and we set each weight component equal to $1/g$. Also, we initialize the parameters b_0^2 and $\Omega^{(0)}$ to ensure the exploration of the entire solution space; to that end, we set each component of b_0^2, for example the i^{th} component $b_{0,i}^2$, to a value so that the range of that parameter is encompassed in the interval $\left(a_{0,i} - 2\sqrt{b_{0,i}^2},\ a_{0,i} + 2\sqrt{b_{0,i}^2}\right)$. Therefore, the entire range is within two sampling standard deviations of the initial mean. We initialize $\Omega^{(0)}$ in the MRAS algorithms in a similar manner, setting it equal to a diagonal matrix with the values of b_0^2 along the diagonal.

We choose the additional parameter values for the CE algorithms as follows (see also Botev and Kroese, 2004): we use a population of $N = 100$ candidates in each iteration, with the number of elite candidates, N^{elite}, equal to 10. The updating parameters for the CE means a and variances b^2 are $\alpha = .9$ and $\beta = .4$, respectively. We choose the additional parameter values for the MRAS algorithms based on Hu et al. (2007): we set $\lambda = .01$, $\epsilon = 10^{-5}$, $\rho_0 = 80$, $N_0 = 100$, and $S(\ell(y, X)) = \exp -\ell(y, X)/1000$. For both the CE and MRAS algorithms, we use the following stopping criterion: $|\zeta_k - \zeta_{k-10}| \leq .1$, where ζ_k is the best log-likelihood value attained in the first k iterations. However, we also run the CE and MRAS algorithms a minimum of 50 iterations to ensure that the algorithms are given enough time to steer away from the initial solution and begin converging to the optimal solution. In other words, the stopping criterion is enforced only after 50 iterations, stopping the methods when no further improvement in the best log-likelihood is attained in the last 10 iterations. Also, we restrict the maximum value of N_k in any iteration of MRAS to be 1000 to limit the computational expense of any single iteration.

3.3.3 Numerical Experiment 1

The first data set consists of 120 points simulated from a 3-mixture bivariate Gaussian distribution; the parameters are displayed in Table 3.4. Notice that this is a relatively simple example with three clusters in the 2-dimensional space; we will use this example to illustrate the different algorithms and their relative performance.

Table 3.5 contains the results of 20 runs of the EM, CE-EM, CE-CD, MRAS-EM, and MRAS-CD algorithms performed on this data set. We report the best

Table 3.4: Parameters used to generate data set 1.

Cluster	Mean	Covariance Matrix	Weight
1	$\begin{pmatrix} 2 & 2 \end{pmatrix}$	$\begin{pmatrix} 1 & .5 \\ .5 & 1 \end{pmatrix}$	.5
2	$\begin{pmatrix} 4 & 8 \end{pmatrix}$	$\begin{pmatrix} .2 & 0 \\ 0 & 1 \end{pmatrix}$	.3
3	$\begin{pmatrix} 0 & 6 \end{pmatrix}$	$\begin{pmatrix} 1 & -.2 \\ -.2 & .5 \end{pmatrix}$	.2

(Max ℓ), worst (Min ℓ), and average (Mean ℓ) solution (log-likelihood value) over the 20 runs. We also report the associated standard error (S.E.(ℓ)) as a measure for the variability of the solutions. Moreover, we report the average number of iterations and the average computing time (in seconds) as a measure for computational effort. And lastly, we report the number of runs M_ϵ^* that come within $\epsilon = .1\%$ of the best solution found. The best solution equals $\ell^* = -413.99$. Since the methods are stochastic, it is unlikely that they all yield the exact same solution. Thus, we consider a solution as approximately equal to the global optimum if it falls within $.1\%$ of ℓ^*.

Table 3.5: Simulation results on data set 1 based on 20 runs.

Algorithm	Max ℓ	Min ℓ	Mean ℓ	S.E.(ℓ)	M_ϵ^*	iters	Avg time
EM	-413.99	-475.14	-431.86	5.36	12	15.20	0.094
CE-EM	-413.99	-414.01	-414.00	0.0016	20	119.85	15.02
CE-CD	-414.03	-414.09	-414.06	0.0038	20	108.65	14.52
MRAS-EM	-414.00	-414.02	-414.01	0.0012	20	101.25	14.25
MRAS-CD	-414.03	-454.22	-416.10	2.01	19	131.85	14.40

The results in Table 3.5 confirm that all of the algorithms have little trouble finding the optimal or near-optimal solutions. The EM algorithm is on the order of

100 times faster than the other methods. However, notice that EM finds the global optimum solution in only 12 out of 20 runs, while our methods are successful in finding the global optimizer every single time (except MRAS-CD, which failed once). In fact, the worst solution (Min) of EM is almost 15% below the global optimum. Although the computational time is somewhat sacrificed, we see that our methods are much more consistent at finding the optimal solution. For instance, the worst solution obtained by our methods is much better than the worst solution obtained by EM; moreover, the variability in the solutions (S.E.(ℓ)) is also much smaller. This is also illustrated pictorially in Figure 3.4, which shows the convergence patterns of all five algorithms. In that figure, we plot the average iteration path of the best solution along with pointwise confidence bounds in the form of plus/minus two times the standard error. The confidence bounds illustrate EM's local convergence behavior: EM gets stuck in local solutions and thus the bounds do not narrow; this is different for the other methods for which, at least eventually, all solutions approach one another.

Figure 3.5 shows typical iteration paths of EM, CE-EM, and CE-CD. We can see that the deterministic nature of EM results in a smooth iteration path until convergence. This is in contrast to the other two methods, where the element of chance can cause uphill moves as well as downhill moves, at least temporarily. For instance, the dips in the iteration path of CE-CD around iterations 75 and 90 are the points where the algorithm injects extra variance into the sampling distribution to increase the search space. Without this injection, the algorithm may prematurely converge to a local maximum; the extra variance increases the search space which can

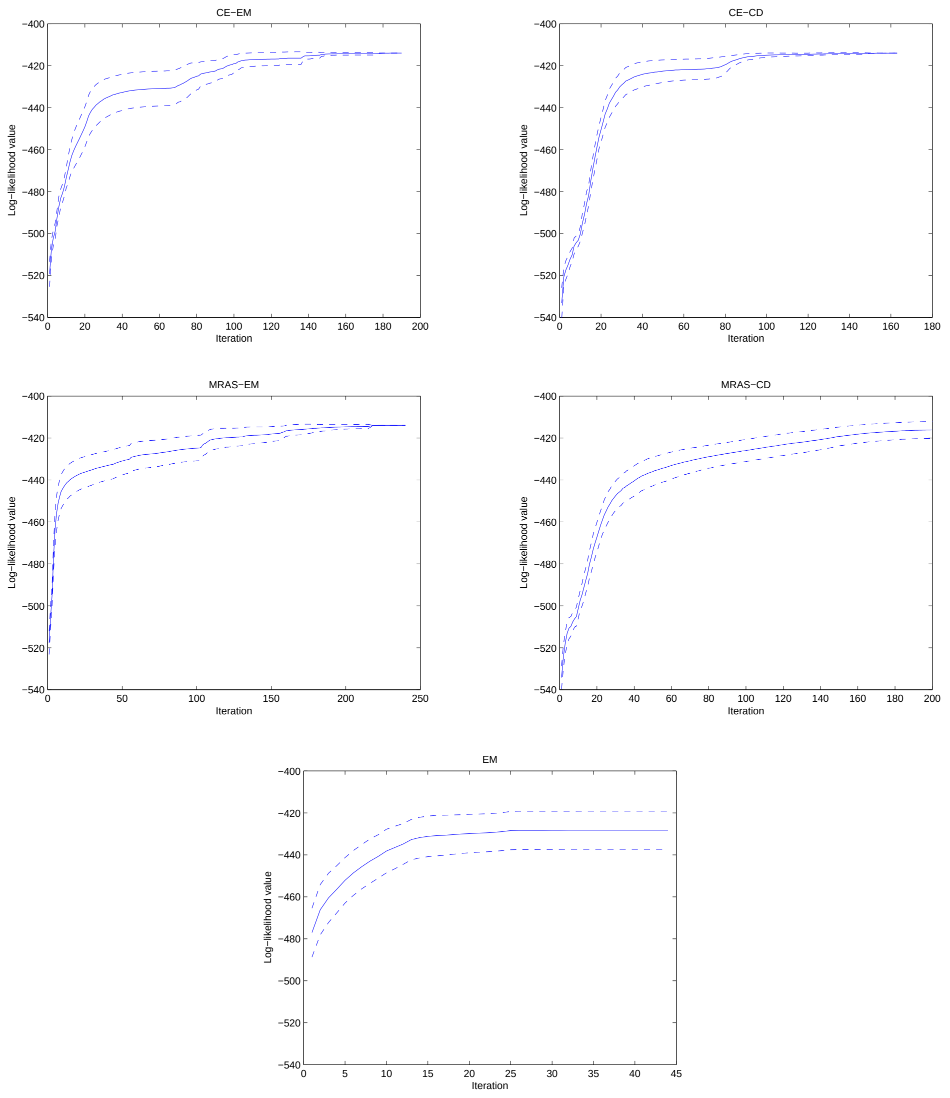

Figure 3.4: Plots of the average iteration paths of the best solution obtained for each method, along with the average plus/minus two times the standard error.

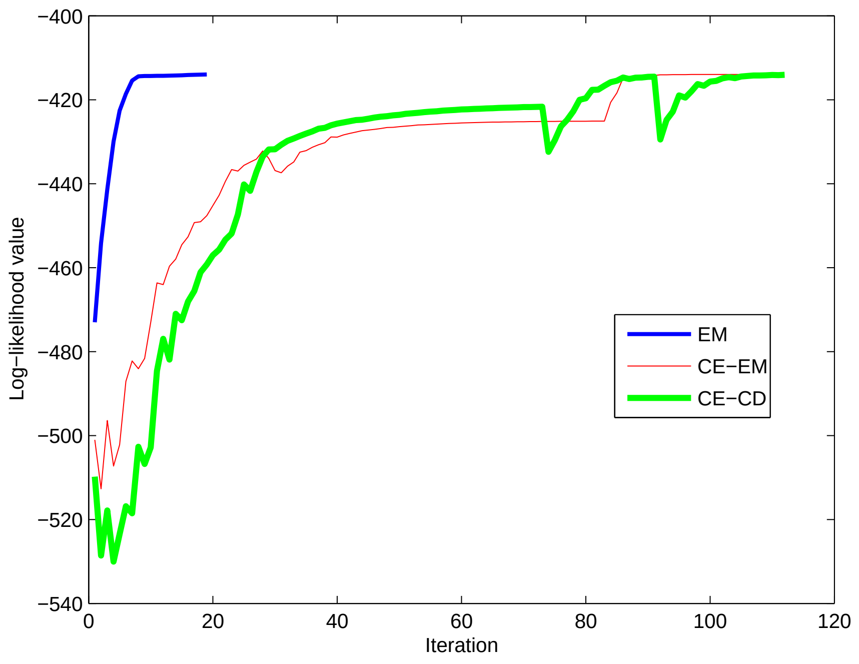

Figure 3.5: Plot of typical iteration paths of EM, CE-EM, and CE-CD on data set 1, where the best log-likelihood value obtained in an iteration is plotted for CE-EM and CE-CD.

steer the algorithm away from the local maximum and toward the global maximum.

In Figures 3.6-3.8, we compare a typical evolution of EM, CE-EM, and CE-CD. Each graph shows two standard deviation ellipses around the estimate for the mean in various iterations as the algorithms evolve. We notice how EM (Figure 3.6) achieves the final (and globally-optimal) solution in fewer iterations. On the other hand, CE-EM (Figure 3.7) spends more computational time searching the solution space before settling on the final solution. This is similar for CE-CD (Figure 3.8). In fact, the covariance matrices for CE-CD converge at a slower pace compared to CE-EM. This is due to the fact that in CE-CD the components for the covariance

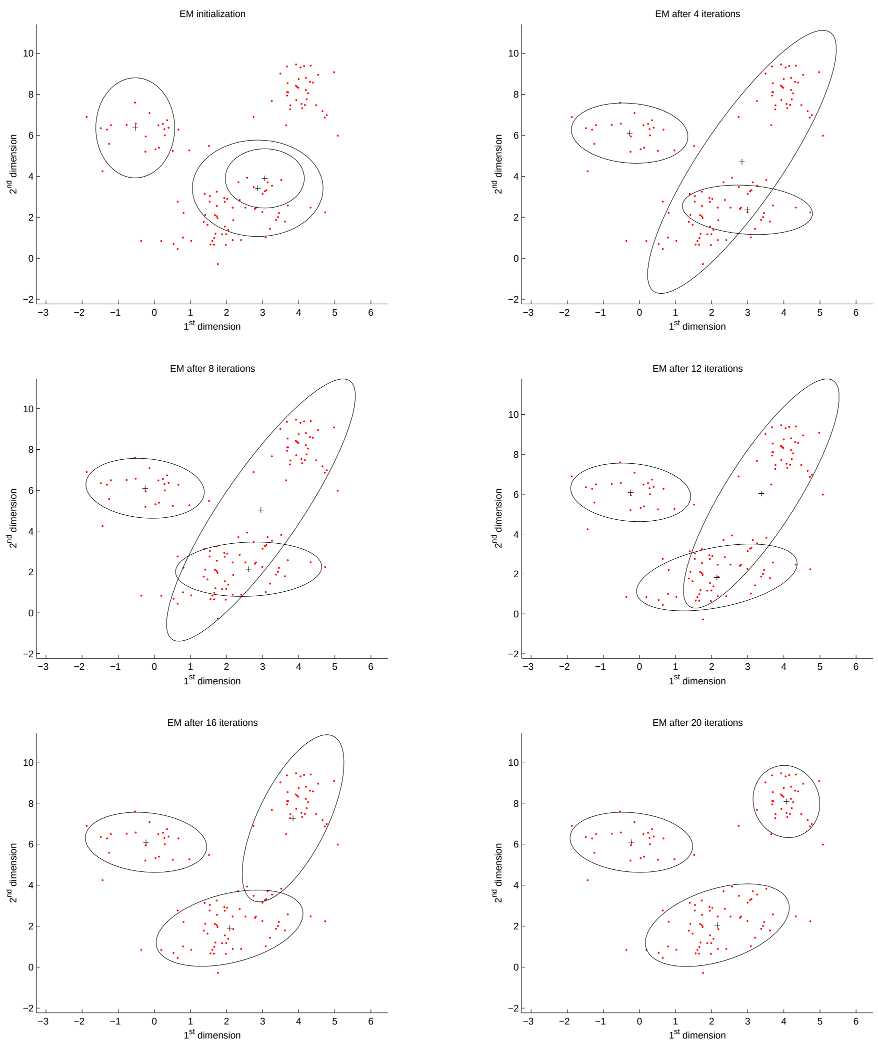

Figure 3.6: Typical evolution of EM on data set 1.

matrices are generated independently of the means and weights, while in EM (and consequently also in CE-EM) this is not the case (see Equations (2.5)-(2.7)). Thus, we can expect convergence of CE-CD to be somewhat slower on average than that of CE-EM. The benefit, though, is the increased search space with respect to the covariance matrices.

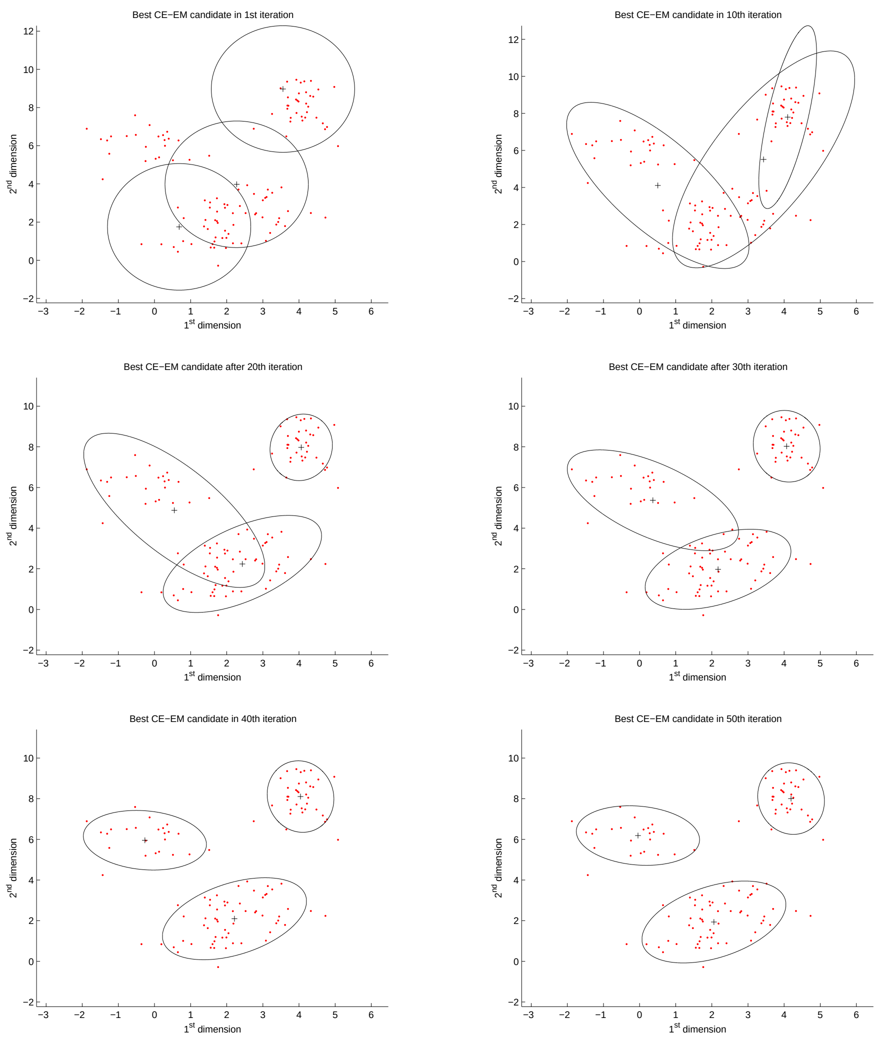

Figure 3.7: Typical evolution of CE-EM on data set 1.

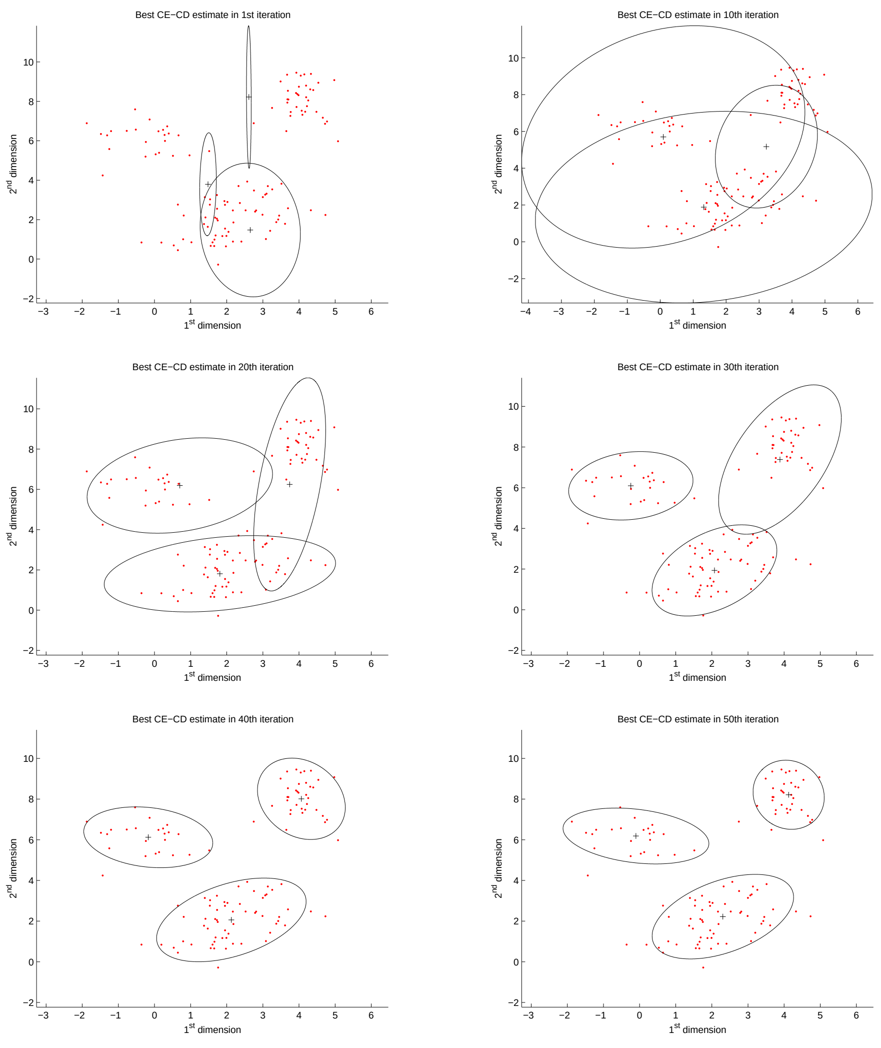

Figure 3.8: Typical evolution of CE-CD on data set 1.

3.3.4 Numerical Experiment 2

For the next data set, we simulate 200 points from a 2-dimensional 6-mixture Gaussian distribution. This data set is similar to the one used in Botev and Kroese (2004), who found that the CE method is superior to EM. Table 3.6 shows the parameters used to generate the data. Notice that this example is much harder than the first one; although we still operate in the two-dimensional space, correctly identifying 6 clusters is much harder than identifying only 3 clusters. Our results will also show that EM has more difficulty in finding the optimal solution.

Table 3.6: Parameters used to generate data set 2.

Cluster	Mean	Covariance Matrix	Weight
1	$\begin{pmatrix} 0.6 & 6 \end{pmatrix}$	$\begin{pmatrix} 1 & .9 \\ .9 & 1 \end{pmatrix}$	.1
2	$\begin{pmatrix} 1 & -10 \end{pmatrix}$	$\begin{pmatrix} 1 & -.9 \\ -.9 & 1 \end{pmatrix}$	.1
3	$\begin{pmatrix} 10 & -1 \end{pmatrix}$	$\begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}$	.2
4	$\begin{pmatrix} 0 & 10 \end{pmatrix}$	$\begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}$	.2
5	$\begin{pmatrix} 1 & -3 \end{pmatrix}$	$\begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}$	.2
6	$\begin{pmatrix} 5 & 5 \end{pmatrix}$	$\begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}$	.2

Table 3.7 shows the results of 20 runs of each method. In that table, we report the percent improvement of the log-likelihood value of the best solution found by each method over the log-likelihood value of the best solution found by EM (imp.), in addition to the values reported in Table 3.5. We see that all four of our methods outperform EM in terms of the best log-likelihood value (Max ℓ). Also,

the worst solution (Min ℓ) is much better than the worst solution found by EM, and the variance of the solutions is smaller. To be fair, we also see that the time for convergence increases drastically for the global optimization methods. However, this may be a fair price for obtaining significantly better solutions.

Table 3.7: Simulation results on data set 2 based on 20 runs.

Algorithm	Max ℓ	Min ℓ	Mean ℓ	S.E.(ℓ)	imp.	iters	Avg time
EM	-956.34	-1060.9	-981.25	5.71	-	23.20	0.26
CE-EM	-947.02	-1019.0	-975.78	4.64	0.97%	257.65	55.50
CE-CD	-946.98	-999.83	-967.73	4.08	0.98%	249.35	56.65
MRAS-EM	-947.08	-1018.4	-979.81	4.10	0.97%	217.15	47.23
MRAS-CD	-947.12	-997.76	-967.92	3.28	0.96%	227.85	58.06

Figure 3.9 shows what can be gained using a global optimization method. In that figure, we see the best solution found by EM versus the best solution found by MRAS-CD. Note that based on Table 3.7 alone, the difference does not appear large (-956.34 for EM versus -947.12 for MRAS-CD, an improvement of less than 1%). However, Figure 3.9 indicates that MRAS-CD identifies the 6 clusters much better than EM. In fact, while both methods correctly identify the bottom 3 clusters, EM has a hard time distinguishing between the upper 3 clusters. While EM overestimates the first cluster, it underestimates the other two. The reason for this poor performance is that the upper 3 clusters are not well separated and EM commits too soon on a final solution. On the other hand, MRAS-CD explores the solution space better and spends a longer time doing so. Only after a thorough exploration of the solution space, MRAS-CD settles on a final solution which also is the true solution.

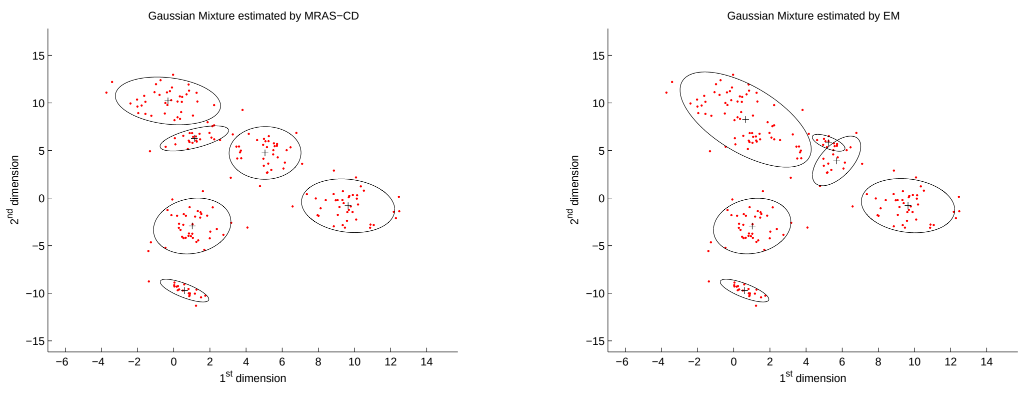

Figure 3.9: The best runs of MRAS-CD and EM on data set 2.

3.3.5 Clustering of Survey Responses

The final data set consists of 152 responses in a survey of MBA students. In that survey, students were asked about their perceived desirability of 10 different cars. The cars ranged from minivans and hatchbacks, to sport utility vehicles and sports cars. Students rated the desirability of each car on a scale of 1-10. Thus, the resulting data set comprises of 152 10-dimensional vectors. The goal of clustering is to find segments in the market of all MBA students with respect to car preferences.

We illustrate our methods on this more challenging data set in the following way. Assuming $g = 3$ clusters, we first standardize the data to make it more amenable to the Gaussian assumption. Because of the larger data-dimesion, we increase the number of candidates generated in each iteration of the CE algorithms, N, to 400, and increase N^{elite} to 20. While this increases the computation time, it allows us a more thorough search of the solution space. We run each method 10 times. Table 3.8 shows the results.

We notice the increased complexity of this problem: the variability of the

Table 3.8: Simulation results on the survey data set based on 10 runs.

Algorithm	Max ℓ	Min ℓ	Mean ℓ	S.E.(ℓ)	imp.	iters	Avg time
EM	-1622.1	-1942.2	-1797.6	28.20	-	13.6	0.14
CE-EM	-1623.4	-1857.2	-1694.8	19.88	-0.08%	151.8	90.50
CE-CD	-1300.6	-1809.9	-1599.3	37.85	19.83%	280.0	265.62
MRAS-EM	-1329.9	-1955.5	-1709.6	39.10	18.01%	117.4	68.45
MRAS-CD	-1435.5	-1886.1	-1620.4	32.87	11.50%	268.4	297.91

solutions is larger and so is the average computing time. However, we also notice the much larger gains of our methods compared to EM: for three out of the four methods, the improvement over EM ranges between 10% and 20%. Only CE-EM fails to produce a better solution than EM. One reason for this underperformance may be the close link to EM via the covariance-updating procedure, which may not allow enough flexibility to explore the entire solution space. We also note that since we increased the number of candidates generated per iteration, the computational time difference between our methods and EM is now even larger. EM is approximately 3 orders of magnitude faster than the new methods in this experiment.

In order to gauge what can be gained from the global optimum, consider the graph in Figure 3.10, which depicts the best set of clusters obtained by each of the two methods, with the data projected onto the first two principal components. We can see that the best solution obtained by MRAS-CD (left panel) separates the data much better into 3 distinct clusters than the best solution obtained by EM (right panel). In particular, in the left panel the cluster means are much better separated, as are the cluster shapes (i.e., the corresponding covariance matrices). The clusters in the left panel span the data set without the amount of overlap seen in the right panel. All-in-all, the cluster assignment corresponding to the best solution obtained by the MRAS-CD algorithm appears to be supported much better by the observed data than that of the best solution obtained by EM.

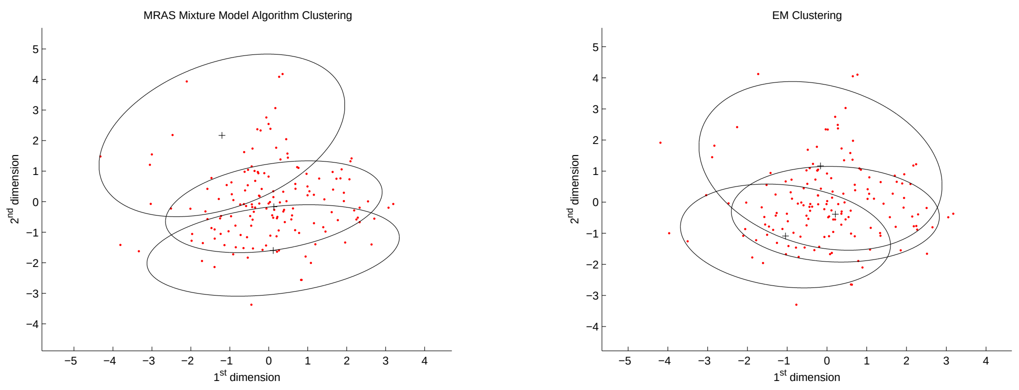

Figure 3.10: The best runs of MRAS-CD and EM on the survey data set, with the data projected onto the first two principal components.

3.3.6 A Fair Comparison

Since EM has considerably faster run-times than the proposed algorithms, one could argue that running EM multiple times with random starting values very well might produce as good or better solutions than a single run of the new algorithms in roughly equivalent time. In the following, we provide a numerical experiment as a fairer comparison between EM and the new algorithms. We use the same data set as in Section 3.3.4, a 200 point, bivariate 6-component Gaussian mixture. However, in this experiment, one simulation includes not just one run of EM, but multiple runs of EM (with random starting values) so that the sum of the run-times totals at least 55 seconds, roughly equal to the run-times of the other algorithms. We report the statistics (Max, Min, Mean, S.E.) of the best runs of EM for each of the 20 simulations.

Table 3.9: Simulation results on data set 2 based on 20 runs for the new algorithms, and 20 sets of multiple runs of EM compliling a total run-time of at least 55 seconds.

Algorithm	Max ℓ	Min ℓ	Mean ℓ	S.E.(ℓ)	Avg time
EM	-946.98	-958.24	-951.04	1.53	55.69
CE-EM	-947.02	-1019.0	-975.78	4.64	55.50
CE-CD	-946.98	-999.83	-967.73	4.08	56.65
MRAS-EM	-947.08	-1018.4	-979.81	4.10	47.23
MRAS-CD	-947.12	-997.76	-967.92	3.28	58.06

Table 3.9 contains the results of the fair comparison simulations. Running EM multiple times for roughly equivalent times compared to the proposed algorithms produces the global optimum on multiple trials. Moreover, the minimum and mean likelihood values across the 20 trials are an improvement over the proposed algorithms. This experiment shows that for the current state of the proposed algorithms and for this particular data set, the algorithms have slightly poorer overall performance on average as multiple runs of EM for equivalent time. However, optimizing the proposed algorithms for speed would reduce the run-times of the algorithms, thus lessening the ability of EM to find the global optimum alloting the same run-times.

3.3.7 Does the Global Optimum "Matter"?

One question that comes to mind when considering local, sub-optimal solutions is whether the global optimum really matters. That is, does knowledge of the global solution make a difference in practice or is the difference merely academic. In the following, we conduct a numerical experiment to answer this question.

In this numerical experiment, we test how the global optimization clustering algorithms perform in relation to EM, not only gauging the goodness of fit by the log-likelihood values, but also using the resulting mixture distribution to classify a select subset of the data points according to the computed clusters. Specifically, we take simulated data, of which we know the underlying parametric distribution as well as the true cluster membership of each data point, and randomly select 70% of the data and declare it the *training set*. We call the remaining 30% of the data the *test set*.

We use the EM, CE-EM, CE-CD, MRAS-EM, and MRAS-CD algorithms to estimate the clusters from the training set, which is held constant throughout the experiment. After we cluster the training set, we use the computed cluster parameters to compute the posterior probabilities of the test set, according to Equation (2.1). Each posterior probability, τ_{ij}, represents the probability that test point i belongs to cluster j. Because we simulated the data, we know the true cluster membership of each data point. We refer to the true membership as τ_{ij}^*, which we set to 1 if test point i was simulated from cluster j, and 0 otherwise. So, we can refer to τ_i^* as the true g-dimensional cluster membership vector for the i^{th} test point.

To quantify how well the the clustering algorithms perform, we look at how the estimated posterior probailities τ_{ij} compare with the true values τ_{ij}^*. One way we do so is to assign each test point y_i to cluster j^*, where $j^* = \text{argmax}_j\, \tau_{ij}$. Then, we count the number of correctly assigned test points and compute the corresponding empirical probabililty $\hat{p}$.

Another way is to compute the norm of the difference of the two vectors τ_i and τ_i^*, effectively computing a distance between the estimated and the true vectors. We compute the average L^2 norm for the data points, which we refer to as $\bar{D}$, via:

$$\bar{D} = \frac{\sum_{i=1}^{n} \|\tau_i - \tau_i^*\|}{n}.$$

We simulate 500 data points from the bivariate Gaussian mixture distribution seen in Table 3.6, and designate 350 of them as the training set, and the remaining 150 as the test set. We then apply each algorithm 20 times. Table 3.10 contains the simulation results of the 20 runs. We report the mean and standard errors of the following values: the resulting log-likelihood values (ℓ), the proportion correctly assigned to the true membership ($\hat{p}$), and the average norm ($\bar{D}$).

Table 3.10: Simulation results on data set 1 based on 20 runs.

Algorithm	Mean ℓ	S.E.(ℓ)	$\hat{p}$	$\sqrt{\hat{p}(1-\hat{p})}$	Mean $\bar{D}$	S.E.($\bar{D}$)
EM	-1759.3	9.31	.810	.0231	.281	.0327
CE-EM	-1749.2	8.34	.808	.0198	.282	.0303
CE-CD	-1721.3	5.66	.911	.0151	.136	.0223
MRAS-EM	-1730.9	4.10	.861	.0168	.201	.0237
MRAS-CD	-1726.1	4.66	.895	.0150	.158	.0221

The results in Table 3.10 indicate that, on average, EM produces solutions with a log-likelihood value of -1759.3, and a standard error of 9.31. On the other hand, all of our four proposed algorithms perform better with average log-likelihood values higher than that of EM. In particular, CE-CD performs the best in this experiment, with an average log-likelihood value of -1721.3. Additionally, the proposed algorithms all produce solutions with lower variability than EM, as evidenced by the smaller standard errors of the likelihood values. EM correctly classifes the test points in this experiment with estimated probability $\hat{p} = .810$, and a standard error of .0231. While the classification rate of CE-EM is marginally lower than EM at .808, the other three algorithms perform significantly better than EM. The two methods that utilize the Cholesky decomposition perform the best, with CE-CD and MRAS-CD correctly classifying 91.1% and 89.5% of the data, an improvement of about 10% over EM. All four proposed algorithms also produce solutions that classify with significantly lower variability than EM, as seen by the lower standard errors of $\hat{p}$. Looking at the average norm $\bar{D}$, again CE-EM has a slightly poorer performance than EM, whose average norm is .281. Analogous to the results for $\hat{p}$, the other three algorithms exhibit much better performance than EM, with CE-CD performing the best with an average of .136. Again, all four algorithms have lower standard errors for the computed norms than EM.

From Table 3.10, we can conclude that the better likelihood values found by CE-CD, MRAS-EM, and MRAS-CD directly translate into better clustering performance on the test set. The only exception is CE-EM, which, despite marginally better (average) solutions, does not cluster the data any better than EM. It is generally also revealing that the CD-based global optimization methods (CE-CD & MRAS-CD) perform better than the ones based on EM. An explanation for this phenomenon is the de-coupling of mean and variance estimation in the CD-based methods. While CE-EM and MRAS-EM use the EM updates for estimating the covariance matrices (which are not independent of the estimated means), CE-CD and MRAS-CD update the covariance matrices independently of the cluster means and cluster proportions. This increased flexibility may lead to an improved exploration of the solution space and, consequently, to better clustering performance.

3.4 Discussion

In this chapter we introduced several new methods to find globally optimal solutions for model-based clustering problems. Numerical experiments indicate that, although they are more computationally intensive than the classical EM, the proposed algorithms perform better on average than EM. In fact, the experiments show that unlike EM, the new methods are relatively insensitive to the starting points. In addition to cross-validating the results using classification techniques as in Section 3.3.7, calculating the likelihood ratio chi-square significance is another way to test the significance of improved solutions for mixture models.

The computational costs of the new methods are orders of magnitudes greater than the EM algorithm, and in some cases, the additional gains in performance are marginal, so an important research avenue to pursue would be a characterization of when the new methods are most effective. Clearly, higher-dimensional problems offer one opportunity, as demonstrated by the MBA survey example example, but even so, it would be worthwhile to determine which properties of the problem lead to more local solutions. Characterizing how the run-time complexity of the proposed algorithms changes with respect to problem size would also be an important factor for determining when the algorithms are most effective.

Another potential avenue of future work for these algorithms would be to generalize the objective function in order to find the best clustering across all values of g. One approach for doing so would be to wrap a suitable model-selection criterion around the algorithms for varying values of g. Other approaches might include incorporating some sort of hierarchical clustering scheme into the proposed algorithms.

Other extensions to our work include application of global optimization methods to general mixture models, as well as other data mining algorithms such as neural networks. Both CE and MRAS provide a natural framework for these optimization problems, and tailoring CE and MRAS to them could result in better global solutions. Also, as the EM algorithm forms the basis for many supervised and unsupervised learning methods, application of these methods to that field are worth investigating.

Chapter 4

Global Convergence of Gaussian Mixture Models with MRAS

4.1 Motivation

Because the likelihood function of Gaussian mixture models typically has a large number of local maxima, finding the global maximum can be a difficult task. Many optimization methods only guarantee convergence to a *local* optimum, and are not necessarily concerned with systematically finding the *global* optimum. In this chapter we discuss a method specifically designed to find the global optimum. The method was first introduced in the field of operations research and is referred to as *Model Reference Adaptive Search*.

Model Reference Adaptive Search (MRAS) is a method that was first proposed in the field of operations research and is designed to attain globally optimal solutions to general multi-extremal continuous optimization problems (Hu et al., 2007). As discussed in Chapter 3, MRAS produces estimates to optimization problems by iteratively generating candidate solutions in each iteration from a parametric sampling distribution. The candidates are all scored according to an objective function, and the highest scoring candidates are used to update the parameters of the sampling distribution by minimizing the Kullback-Leibler (KL) divergence between the sampling distribution and the current reference model. Due to the choice of the reference model sequence, the updating scheme of MRAS leads to a more general framework than the CE method, and allows for rigorous analysis of theoretical convergence (Hu et al., 2007).

Many global optimization algorithms that perform well empirically have no theoretical convergence proofs. A good number of these algorithms are ad-hoc or based on heuristics that do not allow for a rigorous mathematical investigation of their convergence properties. In particular, these approaches lack a built-in mechanism to systematically escape from locally optimal solutions. For instance, the methods we discuss in Chapter 1 that are designed to globally optimize the likelihood function of Gaussian mixture models are not theoretically globally convergent. In contrast, in this chapter we prove global convergence of the MRAS-CD algorithm to the global optimum of Gaussian mixtures. To the best of our knowledge, this is the first mixture analysis algorithm that has provable global convergence. In addition to providing theoretical justification that the algorithm is not merely an ad-hoc heuristic, the convergence proof also gives insight into the performance of the algorithm.

The rest of the chapter begins with an explanation of MRAS in general, as well as some details on its convergence proof in Section 4.2. We discuss the MRAS-CD algorithm and prove its convergence to the global optimum of the likelihood function Gaussian mixture models in Section 4.3. We summarize the findings in the discussion in Section 4.4.

4.2 Model Reference Adaptive Search

Model Reference Adaptive Search (MRAS) is a global optimization tool that estimates the global optimum by generating candidate solutions from a parametric sampling distribution in each iteration. Hu et al. (2007) introduce MRAS as a method that produces solutions to the following global optimization problem:

$$x^* \in \operatorname*{argmax}_{x \in \chi} H(x),\ \ \chi \subseteq \Re^n.$$

MRAS achieves this goal by utilizing a sequence of intermediate reference distributions on the solution space to guide the parameter updates.

The basic methodology of MRAS can be described as follows. In the k^{th} iteration, we generate N_k candidate solutions, $X_1, X_2, ..., X_{N_k}$, according to a sampling distribution $\tilde{g}(\ \cdot\ ; \Upsilon^{(k)})$, where $\Upsilon^{(k)}$ represents the sampling parameters of the k^{th} iteration. After sampling the candidates, we score them according to the objective function, i.e., we compute the objective function value $H(X_i)$ for each candidate X_i. We then obtain an *elite* pool of candidates by selecting the top ρ-percentile scoring candidates. These elite candidates are used to update the parameters of the sampling distribution for the next iteration. An outline of MRAS is given in Figure 4.1.

In MRAS, the value of the percentile ρ changes over the course of the algorithm to ensure that the current iteration's candidates improve upon the candidates in the previous iteration. Let the lowest objective function score among the elite candidates in any iteration k be denoted as γ_k. We introduce a parameter ϵ, a very small positive number, to ensure that the increment in the $\{\gamma_k\}$ sequence is strictly bounded below.

```
  Result: Return highest-scoring estimate for $x^*$
1 Initialize $\Upsilon^{(0)}$.
2 $k \leftarrow 0$.
3 repeat
4     Generate $N_k$ i.i.d. candidate vectors $X_1, ..., X_{N_k}$ from the sampling
      distribution $\tilde{g}(\,\cdot\,; \Upsilon^{(k)})$.
5     Compute the objective function values $H(X_1), H(X_2), ..., H(X_{N_k})$.
6     Select the elite candidates by taking the top scoring $\rho$-percentile
      candidate vectors.
7     Compute the updated MRAS sampling parameters $\Upsilon^{(k+1)}$ via (4.1)
      using the elite candidates.
8     $k \leftarrow k + 1$.
9 until Stopping criterion is met.
```

Figure 4.1: MRAS Outline

If $\gamma_k < \gamma_{k-1}+\epsilon$, increase ρ until $\gamma_k \geq \gamma_{k-1}+\epsilon$, effectively reducing the number of elite candidates. If, however, no such percentile ρ exists, then the number of candidates is increased in the next iteration by a factor of α (where $\alpha > 1$), such that $N_{k+1} = \alpha N_k$. The sampling parameters are then updated as follows:

$$\Upsilon^{(k+1)} := \underset{\Upsilon \in \Theta}{\operatorname{argmax}} \frac{1}{N_k} \sum_{i=1}^{N_k} \frac{[S(H(X_i))]^k}{\tilde{g}(X_i; \Upsilon^{(k)})} I_{\{H(X_i) \geq \gamma_{k+1}\}} \ln g(X_i; \Upsilon), \tag{4.1}$$

where $S : \Re \rightarrow \Re^+$ is a strictly increasing function to account for cases where the objective function value $H(X)$ is negative for a given X, and $I_{\{\cdot\}}$ denotes the indicator function such that:

$$I_{\{A\}} := \begin{cases} 1, & \text{if event A holds,} \\ 0, & \text{otherwise.} \end{cases}$$

The sampling distribution $\tilde{g}$ in MRAS is generally chosen from the natural exponential family. We choose to sample candidate solutions from the Gaussian distribution for our implementation, such that $\Upsilon = (\xi, \Omega)$, where ξ and Ω are the mean and covariance matrix of the MRAS sampling distribution, respectively. The main idea of MRAS is that the sampling parameters will converge to a degenerate distribution centered on the optimal solution; i.e., the sequence of means $\xi^{(0)}, \xi^{(1)}, ...$ will converge to the optimal vector X^*, representing the optimal solution x^*, as the sequence of the sampling covariance matrices $\Omega^{(0)}, \Omega^{(1)}, ...$ converges to the zero matrix. Table 4.1 provides a list of the mixture model parameters and the MRAS parameters.

Table 4.1: List of the model and MRAS parameters.

Mixture Model parameters	MRAS parameters
n = number of data points y_i = i^{th} data point p = dimension of data g = number of mixture components π_j = weight of j^{th} mixture component ψ_j = probability distribution parameters of j^{th} component ψ_j = probability distribution parameters of j^{th} component $f_j(\,\cdot\,;\psi_j)$ = probability density of j^{th} component θ = model parameters to estimate θ^* = model parameters that represent the global optimum $\ell(y,\theta)$ = log-likelihood function μ_j = Gaussian mixture mean vector Σ_j = Gaussian mixture covariance matrix	$\Upsilon^{(k)}$ = sampling parameters in k^{th} iteration $\tilde{g}(\,\cdot\,;\Upsilon^{(k)})$ = sampling density N_k = number of candidates in k^{th} iteration X_i = candidate vector ρ = elite candidate percentile γ_k = lowest objective score of elite candidates in k^{th} iteration λ = sampling weight $S : \Re \rightarrow \Re^+$ = strictly increasing function X^* = candidate vector representing the global optimum ϵ = lower bound on the increase of each γ_k $\xi^{(k)}$ = Gaussian sampling mean vector $\Omega^{(k)}$ = Gaussian sampling covariance matrix χ = constrained domain for candidate vectors $H(\,\cdot\,)$ = objective function Θ = constrained domain for sampling parameters

While MRAS is generally very versatile, applying it to the mixture model context is not straightforward. Part of the reason is that the mixture model requires simulation of candidate solutions that satisfy the mixture model constraints. In the following, we propose a solution via the Cholesky decomposition in order to assure an efficient implementation of MRAS.

4.2.1 Global Convergence of MRAS

In this section we discuss the global convergence properties of MRAS mixture model algorithm in the finite mixture model problem. We must first revert to the general MRAS framework, where Hu et al. (2007) provide a convergence proof of MRAS to the globally optimal solution when using a sampling distribution $g(\,\cdot\,;\Upsilon)$ that belongs to the exponential family. Before we discuss the theorem, we provide the definition of the exponential family of distributions, as well as some required assumptions for the theorem.

Definition 4.1 *A parameterized family of p.d.f.'s* $\{g(\,\cdot\,;\Upsilon), \Upsilon \in \Theta \subseteq \Re^m\}$ *on* χ *is said to belong to the exponential family if there exists functions* $h : \Re^n \to \Re, \Gamma : \Re^n \to \Re^m$, *and* $K : \Re^m \to \Re$ *such that*

$$g(x;\Upsilon) = \exp\{\Upsilon^T \Gamma(x) - K(\Upsilon)\}h(x), \ \forall \Upsilon \in \Theta,$$

where $K(\Upsilon) = \ln \int_{x\in\chi} \exp\{\Upsilon^T \Gamma(x)\}h(x)dx$.

The following assumptions are referenced in the statement of Theorem 4.1.

Assumptions:

A1. For any given constant $\xi < H(x^*)$, the set $\{x : H(x) \geq \xi\} \cap \chi$ has a strictly positive Lebesgue measure.

A2. For any given constant $\delta > 0$, $\sup_{x \in A_\delta} H(x) < H(x^*)$, where $A_\delta := \{x : \|x - x^*\| \geq \delta\} \cap \chi$ and the supremum over the empty set is defined to be $-\infty$.

A3. There exists a compact set Π_ϵ such that $\{x : H(x) \geq H(x^*) - \epsilon\} \cap \chi \subseteq \Pi_\epsilon$. Moreover, $g(x; \Upsilon^{(0)})$ is bounded away from zero on Π_ϵ, i.e., $g_* = \inf_{x \in \Pi_\epsilon} g(x; \Upsilon^{(0)}) > 0$.

A4. The parameter vector $\Upsilon^{(k)}$ computed in (4.1) is an interior point of Θ for all k.

In Theorem 4.1, Hu et al. (2007) show global convergence of MRAS to the optimal solution x^* when using the multivariate Gaussian sampling distribution. As the number of iterations tends to infinity, the sampling distribution tends toward a degenerate distribution centered on the optimal solution x^*.

Theorem 4.1 *If multivariate Gaussian p.d.f.'s are used in MRAS, i.e.,*

$$g(X; \xi^{(k)}, \Omega^{(k)}) = \frac{1}{\sqrt{(2\pi)^n |\Omega^{(k)}|}} \exp\left(-\frac{1}{2}(X - \xi^{(k)})^T (\Omega^{(k)})^{-1} (X - \xi^{(k)})\right),$$

$\epsilon > 0, \alpha > (\beta S^*)^2$, *and Assumptions A1, A2, A3, and A4 are satisfied, then*

$$\lim_{k \to \infty} \xi^{(k)} = x^*, and \lim_{k \to \infty} \Omega^{(k)} = 0_{n \times n} \; w.p. \; 1$$

4.3 MRAS algorithm for Gaussian Mixture Models

As pointed out above, MRAS requires, in every iteration, the simulation of candidate solutions from within the parameter space. In the Gaussian mixture model, these candidate solutions must include the mixture weights $\pi = (\pi_1, ..., \pi_g)$ and the probability distribution parameters $\psi_j = (\mu_j, \Sigma_j)$ for $j = 1, ..., g$, where μ_j is the mean vector and Σ_j is the covariance matrix of the j^{th} component. Simulating covariance matrices is involved, since they need to be positive definite. Naive approaches (e.g., via simulating matrices randomly and consequently selecting only those that are positive definite) can be extremely inefficient. In Chapter 3 the MRAS-CD algorithm was introduced as an algorithm utilizes updating the Cholesky factorization of a covariance matrix to efficiently simulate s.p.d. covariance matrices. that applied MRAS to Gaussian mixture models updated the covariance matrices of iteratively. In this chapter, we look more closely at one of those algorithms, namely MRAS-CD. and we proposed several algorithm. In th following, we propose a new method to simulate positive definite covariance matrices for the MRAS mixture model algorithm. This method relies on the Cholesky decomposition. Recall the following theorem (see e.g., Thisted, 1988) regarding the Cholesky decomposition of a symmetric positive definite matrix:

Theorem 4.2 *A real, symmetric matrix A is symmetric positive definite (s.p.d.) if and only if it has a Cholesky decomposition such that $A = U^T U$, where U is a real-valued upper triangular matrix.*

Because covariance matrices are s.p.d., each covariance matrix has a corresponding Cholesky factorization U. Therefore, one way to stochastically generate covariance matrices in the MRAS mixture model is to generate the components of the U matrix from the Cholesky decomposition instead of the components of the covariance matrix Σ directly. Note that only the $p(p+1)/2$ upper right-hand components of U must be generated for each $p \times p$ covariance matrix (all other components are necessarily zero). Then the covariance matrix can be constructed from the simulated Cholesky factors, ensuring that the covariance matrix is s.p.d.

One potential problem with this method is that the Cholesky factorization for a symmetric positive definite matrix is not unique. For a Cholesky factorization U of Σ, we can multiply any subset of rows of U by -1 and obtain a different Cholesky factorization of the same Σ. Thus, there is not a unique global optimum in the MRAS mixture model algorithm. However, in their discussion of parameterizations of positive definite matrices, Pinheiro and Bates (1996) note that if the diagonal elements of the Cholesky factorization U are required to be positive, then the Cholesky factorization U is unique. Thus, by restricting the diagonal elements of U to be positive, we can circumvent the uniqueness problem of the Cholesky factorization mentioned above. We therefore choose to construct the covariance matrices in the MRAS mixture model algorithm by sampling the diagonal components of U from a truncated Gaussian distribution (accepting all positive values), and subsequently computing the covariance matrix $\Sigma = U^T U$.

MRAS can now be applied to the estimation of Gaussian mixtures in the following way. We first sample candidate solutions X_i that correspond to the set of mixture parameters $\theta = (\mu_j, \Sigma_j, \pi_j)_{j=1}^{g}$, where the covariance matrices are represented by their corresponding Cholesky factorizations mentioned above. We then score each candidate with the log-likelihood function, and use the best-scoring candidates to update the sampling distribution. The goal is to obtain the optimal solution X^* containing the mixture means, Cholesky factorizations, and weights of the *global* maximum likelihood estimate $\theta^* = (\mu_j^*, \Sigma_j^*, \pi_j^*)_{j=1}^g$. We provide the MRAS mixture model algorithm in Figure 4.2. Note that the MRAS parameter λ is a small constant which assigns a probability to sample from the initial sampling distribution $g(\,\cdot\,; \Upsilon^{(0)})$ in any subsequent iteration. Also, $g(\,\cdot\,; \xi, \Omega)$ is the multivariate Gaussian density, i.e.,

$$g(X; \xi, \Omega) = \frac{1}{\sqrt{(2\pi)^p |\Omega|}} \exp\left(-\frac{1}{2}(X - \xi)^T \Omega^{-1} (X - \xi) \right),$$

where $|\Omega|$ denotes the determinant of the matrix Ω.

The stopping criterion for the MRAS mixture model algorithm that we use is to stop when the increase of the best log-likelihood value over k iterations falls below a specified tolerance.

4.3.1 Preventing Degenerate Solutions

As mentioned in Section 3.3.1, maximizing the log-likelihood function in the Gaussian mixture model can lead to unbounded solutions, if the parameter space is not properly constrained. Therefore, we choose to constrain the MRAS mixture model algorithm generates a covariance matrix that violates the constraint given by Equation (3.3), we discard the candidate and re-generate a new one.

```
Data: Data points $y_1, y_2, ..., y_n$
Result: Return highest-scoring estimate for $X^*$.
Initialize $\xi^{(0)}$ and $\Omega^{(0)}$.
$k \leftarrow 0$.
repeat
    Generate $N_k$ i.i.d. candidate vectors $X_1, ..., X_{N_k}$ from the sampling
    distribution $\tilde{g}(\,\cdot\,; \xi^{(k)}, \Omega^{(k)}) := (1-\lambda)g(\,\cdot\,; \xi^{(k)}, \Omega^{(k)}) + \lambda g(\,\cdot\,; \xi^{(0)}, \Omega^{(0)})$.
    Compute the log-likelihoods values $\ell(y, X_1), \ell(y, X_2), ..., \ell(y, X_{N_k})$.
    Select the elite candidates by taking the top scoring $\rho_{k-1}$-percentile
    candidate vectors, and define $\tilde{\gamma}_k(\rho_k)$ as the $\rho_k$-percentile log-likelihood
    score obtained of all candidates in iteration $k$.
    if $k = 0$ or $\tilde{\gamma}_k(\rho_k) \geq \gamma_k + \frac{\epsilon}{2}$ then
        $\gamma_{k+1} \leftarrow \tilde{\gamma}_k(\rho_k), \rho_{k+1} \leftarrow \rho_k$, and $N_{k+1} \leftarrow N_k$.
    else
        find the largest $\tilde{\rho} \in (\rho_k, 100)$ such that $\tilde{\gamma}_k(\tilde{\rho}) \geq \gamma_k + \frac{\epsilon}{2}$.
        if such a $\tilde{\rho}$ exists then
            $\gamma_{k+1} \leftarrow \tilde{\gamma}_k(\tilde{\rho})$, $\rho_{k+1} \leftarrow \tilde{\rho}$, and $N_{k+1} \leftarrow N_k$,
        else
            $\gamma_{k+1} \leftarrow \gamma_k$, $\rho_{k+1} \leftarrow \rho_k$, and $N_{k+1} \leftarrow \alpha N_k$.
        end
    end
    Update the sampling parameters according to:

        $\xi^{(k+1)} \leftarrow \frac{\sum_{i=1}^{N_k} \frac{S(\ell(y,X_i))^k X_i}{\tilde{g}(X_i,\xi^{(k)},\Omega^{(k)})} I_{\{\ell(y,X_i)\geq\gamma_{k+1}\}}}{\sum_{i=1}^{N_k} \frac{S(\ell(y,X_i))^k}{\tilde{g}(X_i,\xi^{(k)},\Omega^{(k)})} I_{\{\ell(y,X_i)\geq\gamma_{k+1}\}}}, \quad (4.2)$

        $\Omega^{(k+1)} \leftarrow \frac{\sum_{i=1}^{N_k} \frac{S(\ell(y,X_i))^k (X_i-\xi^{(k+1)})(X_i-\xi^{(k+1)})^T}{\tilde{g}(X_i,\xi^{(k)},\Omega^{(k)})} I_{\{\ell(y,X_i)\geq\gamma_{k+1}\}}}{\sum_{i=1}^{N_k} \frac{S(\ell(y,X_i))^k}{\tilde{g}(X_i,\xi^{(k)},\Omega^{(k)})} I_{\{\ell(y,X_i)\geq\gamma_{k+1}\}}}. \quad (4.3)$

    $k \leftarrow k + 1$.
until Stopping criterion is met.
```

Figure 4.2: MRAS Mixture Model Algorithm

4.3.2 Proving Global Convergence of the MRAS Mixture Model Algorithm

In order to show that Theorem 4.1 applies to the MRAS mixture model algorithm algorithm, we must show that Assumptions A1, A2, A3, and A4 hold true in the maximization of the likelihood function of the mixture density. So, for our purposes, the objective function $H(x)$ discussed in the general MRAS framework is the log-likelihood of the mixture density:

$$\ell(y,\theta) = \sum_{i=1}^{n} \log \sum_{j=1}^{g} \pi_j f_j(y_i; \mu_j, \Sigma_j).$$

In the MRAS mixture model algorithm, we are estimating the vector X^* representing the optimal means, weights, and Cholesky factorizations of the covariance matrices, i.e., the vector X^* representing the optimal solution $\theta^* = (\mu_i^*, \Sigma_i^*, \pi_i^*)_{i=1}^{g}$. Therefore, we are trying to solve the optimization problem:

$$X^* \in \operatorname*{argmax}_{X \in \chi} \ell(y, X).$$

Before we prove the global convergence of the MRAS mixture model algorithm, we first provide the following useful lemmas. Lemma 4.1 shows that a continuous function that is bounded above and possesses a unique optimal maximizer on a constrained space $\chi \in \Re^n$ satisfies Assumption A1.

Lemma 4.1 *For a continuous function $H(x)$, $x \in \chi \in \Re^n$, where H is bounded above and there exists a unique optimal maximizer x^* s.t. $H(x) < H(x^*)$, $\forall x \neq x^*$, then $\forall \xi < H(x^*)$, the set $\{x : H(x) \geq \xi\}$ has strictly positive Lebesgue measure, and thereby Assumption A1 is satisfied.*

Proof: Choose $\xi < H(x^*)$ and let $\epsilon = H(x^*) - \xi > 0$. By continuity of H, $\exists \delta > 0$ s.t. $\forall x \in \{x : \|x - x^*\| < \delta\}$, then $|H(x) - H(x^*)| < \epsilon$. By rewriting the left- and right-hand sides of the inequality, we see that $H(x^*) - H(x) < H(x^*) - \xi$, i.e., $\xi < H(x)$, $\forall x \in \{x : \|x - x^*\| < \delta\}$. Since the set $\{x : \|x - x^*\| \leq \frac{\delta}{2}\} \subseteq \{x : H(x) \geq \xi\}$, then $m\left(\{x : H(x) \geq \xi\}\right) \geq m\left(\{x : \|x - x^*\| \leq \frac{\delta}{2}\}\right) > 0$. $\square$

Lemma 4.2 gives an inequality relating the determinants of two positive definite $n \times n$ matrices with the determinant of their convex combination (see e.g., Horn and Johnson, 1990).

Lemma 4.2 *For positive definite $n \times n$ matrices A and B,*

$$\det\left(\alpha A + (1-\alpha)B\right) \geq (\det A)^{\alpha}(\det B)^{1-\alpha}, \text{ where } \alpha \in (0,1).$$

In Lemma 4.3 we extend the statement of Lemma 4.2 to a convex combination of an arbitrary number of positive definite $n \times n$ matrices. In the proof, we make use of two properties of positive definite matrices: for positive definite matrices A, B, and scalar $\alpha > 0$, then αA and $A + B$ are both positive definite as well (Johnson, 1970).

Lemma 4.3 *For positive definite $n \times n$ matrices A_j, $j = 1, ..., k$,*

$$\det\left(\sum_{j=1}^{k} \alpha_j A_j\right) \geq \prod_{j=1}^{k} (\det A_j)^{\alpha_j},$$

for any set of $\{\alpha_j\}_{j=1}^{k}$ s.t. $\alpha_j > 0$ and $\sum_{j=1}^{k} \alpha_j = 1$.

Proof: We prove this lemma by induction.

i. Base case: $k = 2$, shown by Lemma 4.2.

ii. Assuming the lemma holds for k, we show it holds for $k + 1$, i.e., for any set $\{\tilde{\alpha}_j\}_{j=1}^{k+1}$ s.t. $\tilde{\alpha}_j > 0$ and $\sum_{j=1}^{k+1} \tilde{\alpha}_j = 1$, then,

$$\det\left(\sum_{j=1}^{k+1} \tilde{\alpha}_j A_j\right) \geq \prod_{j=1}^{k+1} (\det A_j)^{\tilde{\alpha}_j}.$$

Define $\alpha_j = \frac{\tilde{\alpha}_j}{1-\tilde{\alpha}_{k+1}}$, for $j = 1, ..., k$. Thus, $\sum_{j=1}^{k} \alpha_j = 1$ and the induction assumption can be applied to $\{A_1, ..., A_k\}$ for this set of $\{\alpha_j\}_{j=1}^{k}$. Then,

$$\begin{aligned}
\det\left(\sum_{j=1}^{k+1} \tilde{\alpha}_j A_j\right) &= \det\left(\sum_{j=1}^{k} \tilde{\alpha}_j A_j + \tilde{\alpha}_{k+1} A_{k+1}\right) \\
&= \det\left((1-\tilde{\alpha}_{k+1}) \sum_{j=1}^{k} \alpha_j A_j + \tilde{\alpha}_{k+1} A_{k+1}\right) \\
&\geq \left[\det\left(\sum_{j=1}^{k} \alpha_j A_j\right)\right]^{1-\tilde{\alpha}_{k+1}} (\det A_{k+1})^{\tilde{\alpha}_{k+1}} \text{ (by Lemma 2)} \\
&\geq \left[\prod_{j=1}^{k} (\det A_j)^{\alpha_j}\right]^{1-\tilde{\alpha}_{k+1}} (\det A_{k+1})^{\alpha_{\widetilde{k+1}}} \quad \begin{pmatrix}\text{by induction} \\ \text{assumption}\end{pmatrix} \\
&= \left[\prod_{j=1}^{k} (\det A_j)^{\tilde{\alpha}_j}\right] (\det A_{k+1})^{\tilde{\alpha}_{k+1}} \\
&= \prod_{j=1}^{k+1} (\det A_j)^{\tilde{\alpha}_j}.
\end{aligned}$$

Therefore, we have shown by induction that the statement of the lemma is true. □

Constraining the parameter space is necessary for the proof of the MRAS mixture model algorithm convergence theorem. As mentioned in Section 3.3, we must place additional constraints on the parameter space in order to prevent degenerate clusters and an unbounded log-likelihood value. Specifically, these constraints are $|U_j^T U_j| \geq c > 0$, $j = 1, ..., g$, i.e., bounding the generalized variances of the covariance matrices below. We simplify this constraint by relying on a convenient property of determinants of positive definite matrices: for positive definite A, B, $\det AB = \det A \det B$. So, for the Cholesky decomposition $\Sigma = U^T U$, $|\Sigma| = |U^T||U| = |U|^2$. Equivalently, we write $|U| \geq \sqrt{c}$. Since U is an upper-triangular matrix, $|U|$ is equal to the product of its diagonal elements. So, the constraint $|U^T U| \geq c$ can be written as $\prod_{i=1}^{p} U_{ii} \geq \sqrt{c}$.

One condition that is necessary for satisfying Assumption A2 is that the optimal candidate solution X^* be unique. By restricting the diagonal components of the Cholesky factorization U to be positive, its correponding covariance matrix Σ is unique. However, for a given optimal solution, any permutation of the cluster labels will result in an equivalent log-likelihood value to the problem, resulting in $g!$ optimal solutions and therefore a non-indentifiable formulation. To avoid this problem, we add the following constraint to the problem:

$$\mu_1(1) \leq \mu_2(1) \leq \mu_3(1) \leq ... \leq \mu_g(1),$$

where $\mu_i(1)$ represents the 1^{st} mean component of the i^{th} cluster. Although the inequalities in this constraint are not strict, the probability of multiple mean components of continuous random data being equal is zero. Therefore, this constraint mandates a unique ordering of the mixture components of θ^* w.p. 1 for continuous random data, resulting in a unique optimal candidate solution X^* to the MRAS Gaussian mixture model algorithm.

To allow us to prove convergence, we choose to bound the candidate means within a compact space based on the observed data set. In particular, we define y_{min} as the minimum value over all components of the data points $y_1, ..., y_n$. That is, $y_{min}(i) = \min_{j=1,...,n} y_j(i)$. Similarly, we define y_{max} as the maximum value over all components of the data points, i.e., $y_{max}(i) = \max_{j=1,...,n} y_j(i)$. We note that bounding the candidate mean components by y_{min} and y_{max} is not an unreasonable constraint; clearly, the means of the optimal clusters will not lie outside the range of the data.

We also place constraints on the components of the Cholesky factorizations when we generate the candidate vectors. We first calculate the sample variance of the data set, $Var(\{y_1, y_2, ..., y_n\})$, and then choose the maximum across all p components, i.e., $V_{max} = \max_{i=1,...,p} Var(\{y_1, y_2, ..., y_n\})$. And so, V_{max} represents an upper bound for the variance component of any cluster. Constraining the diagonal components of the Cholesky factorizations within the bounds $[0, V_{max}]$ and the off-diagonal non-zero components within $[-V_{max}, V_{max}]$ suffices, as the global optima will undeniably satisfy these constraints.

Therefore, the solution space with all of the necessary constraints is given by the following:

$$\chi = \begin{cases} \mu_j \in [y_{min}, y_{max}], & j = 1, ..., g \\ \quad \text{s.t. } \mu_1(1) \le \mu_2(1) \le ... \le \mu_g(1) & \\ U_{j(ii)} \in [0, V_{max}], & j = 1, ..., g; \; i = 1, ..., p \\ \quad \text{s.t. } \prod_{i=1}^{p} U_{j(ii)} \ge \sqrt{c} > 0, & j = 1, ..., g \\ U_{j(ik)} \in [-V_{max}, V_{max}], & j = 1, ..., g; \; i = 1, ..., p-1; \\ & \quad k = i+1, ..., p \\ \pi_j \in [0, 1], & j = 1, ..., g \\ \quad \text{s.t. } \sum_{j=1}^{g} \pi_j = 1 & \end{cases} \tag{4.4}$$

The number of parameters that we are estimating, namely the means, weights, and the upper-triangular entries of the Cholesky factorization (all other components are necessarily zero) for each cluster, is $d := g\frac{(p+1)(p+2)}{2}$, so we can consider the space χ to be d-dimensional. The MRAS sampling parameters $\Upsilon = (\xi, \Omega)$ belong to the space Θ, where $\Theta = \{\xi \in \chi,\ \Omega \text{ is s.p.d.}\}$.

Lemma 4.4 *The subspace $\chi \subseteq \Re^d$ is compact.*

Proof: For any vector $X \in \chi$, we note that all components of X are bounded as described in (4.4). We now show that the space χ is closed. The constraints that bound the space clearly constitute a closed subspace in $\Re^d$. The remaining constraints, namely $\mu_1(1) \leq \mu_2(1) \leq ... \leq \mu_g(1)$, $\prod_{i=1}^{p} U_{j(ii)} \geq \sqrt{c}$ for $j = 1, ..., g$, and $\sum_{j=1}^{g} \pi_j = 1$, each represent a closed subspace of $\Re^d$, because all inequalities on the constraints are not strict. Therefore, χ is a finite intersection of closed sets, and is thus closed. Because χ is both closed and bounded, then χ is compact. $\square$

Lemma 4.5 *For a continuous function $H(x)$, $x \in \chi \in \Re^n$, where H is bounded above and there exists a unique optimal solution x^* s.t. $H(x) < H(x^*)$, $\forall x \neq x^*$ and χ is a compact space, then $\forall \delta > 0$, $\sup_{x \in A_\delta} H(x) < H(x^*)$, where $A_\delta := \{x : \|x - x^*\| \geq \delta\} \cap \chi$, and thereby Assumption A2 is satisfied.*

Proof: We prove this lemma directly:

We can rewrite $A_\delta = \chi \setminus \{x : \|x - x^*\| < \delta\}$, which is the complement of the open ball of radius δ around x^* intersected with χ. Therefore, since χ is a compact space, A_δ is a compact space as well.

Since $H(x)$ is a continuous function, it achieves its supremum on the compact space A_δ, i.e., $\exists \tilde{x} \in A_\delta$ s.t. $\sup_{x \in A_\delta} H(x) = H(\tilde{x})$.

And, because $H(x) < H(x^*)$, $\forall x \neq x^*$, we have:

$$\sup_{x \in A_\delta} H(x) = H(\tilde{x}) < H(x^*). \quad \square$$

Now we give Theorem 4.3, where we show that Theorem 4.1 applies to MRAS mixture model algorithm in the global optimization of Gaussian finite mixture models.

Theorem 4.3 *For the maximization of the likelihood function of a mixture density of g Gaussian clusters, if the MRAS parameters are chosen s.t. $\epsilon > 0, \alpha > (\beta S^*)^2$, where $S^* := S(\ell(y, \theta^*))$, and we are optimizing over the compact space χ denoted by (4.4), then:*

$$\lim_{k \to \infty} \xi^{(k)} = X^*, \textit{and} \lim_{k \to \infty} \Omega^{(k)} = 0_{d \times d} \ \textit{w.p.}\ 1,$$

where X^* is the unique optimal vector representing the MLE $\theta^* = (\mu_j^*, \Sigma_j^*, \pi_j^*)_{j=1}^g$.

Proof: This proof consists of showing that Assumptions A1, A2, A3, and A4 apply to MRAS mixture model algorithm in the maximization of the log-likelihood of the Gaussian mixture density.

i. Because $\ell(y, \theta)$ is continuous on χ w.r.t. θ, then by Lemma 4.1, for any $\xi < \ell(y, \theta^*)$, the set $\{y : \ell(y, \theta) \geq \xi\} \cap \chi$ has a strictly positive Lebesgue measure. Thus, Assumption A1 is satisfied.

ii. By Lemma 4.5, since $\ell(y, \theta)$ is continuous on χ w.r.t. θ, then $\forall \delta > 0$, $\sup_{\theta \in A_\delta} \ell(y, \theta) < \ell(y, \theta^*)$, where $A_\delta := \{\theta : \|\theta - \theta^*\| \geq \delta\} \cap \chi$. And so, Assumption A2 is satisfied.

iii. By restricting the search space to a compact region, then the set $\{\theta : \ell(y,\theta) \geq \ell(y,\theta^*) - \epsilon\} \cap \chi$ is a subset of a compact set, namely χ itself. Moreover, using a multivariate Gaussian sampling distribution ensures that sampling any point in the entire solution space on the first iteration occurs with non-zero probability. Thus, A3 is shown.

iv. In order to show that the formulation satisfies A4, we first revisit the updating scheme of MRAS when the sampling distribution is multivariate Gaussian as given by Equations (4.2) and (4.3). It is evident that the mean of the sampling distribution, $\xi^{(t)}$, is simply a convex combination of the elite candidates. Since each candidate $X_i \in \chi$, then a convex combination of them will satisfy all of the constraints as well. One can verify this by noting that the space χ is convex; this is clearly evident for all of the constraints in the formulation, except for the degenerate cluster constraint, $|U_j| \geq \sqrt{c} > 0,\ j = 1, ..., g$, which we now address.

    We need to show that a convex combination of the top t candidates also satisfies this constraint, namely $\left|\sum_{j=1}^{t} \alpha_j U_j\right| \geq \sqrt{c}$. We note that as a direct application of Lemma 4.3, $\left|\sum_{j=1}^{t} \alpha_j U_j\right| \geq \prod_{j=1}^{t} |U_j|^{\alpha_j} \geq \min_j |U_j| \geq \sqrt{c}$. This shows that a convex combination of Cholesky factorizations satisfying the degenerate constraint will also satisfy the degenerate constraint.

    Also, because the candidates X_i are sampled from the probability distribution $\tilde{g}(\,\cdot\,; \xi^{(k)}, \Omega^{(k)})$, then w.p. 1 each candidate lies in the interior of χ. Therefore, the updated mean vector $\xi^{(k+1)}$ will also lie in the interior of the space. Also, the updated $\Omega^{(k+1)}$ is clearly s.p.d. by construction, and thus A4 is satisfied.

$\square$

4.4 Discussion

In this chapter we presented a proof of global convergence of the MRAS-CD algorithm to the optimal solution for Gaussian mixtures. In addition to its theoretical convergence, the numerical experiments discussed in Chapter 3 indicate that the proposed algorithm can find better global solutions missed by the classical EM. Furthermore, we note that by restricting the parameter space of the optimization decision variables to a compact set, the proof can be extended to finite mixture models of other probability distributions as well, in addition to Gaussian as presented in this chapter.

Chapter 5

Landscape Analysis of Finite Mixture Models

5.1 Motivation

In this chapter we examine the likelihood function of finite mixture models, specifically focusing on metrics that measure the difficulty of finding the global optimum of a given data set. Note that although the standard formulation of finite mixture models as given in Chapter 2 is a non-identifiable formulation due to the label-switching problem addressed in Chapter 4, mandating a unique ordering of the labels results in a unique global optimum for mixture models. This dissertation has focused on global optimization of mixture models, and now in this chapter we focus on some of the underlying reasons for why optimizing the likelihood function of mixture models is difficult. We do so by analyzing the likelihood function's fitness landscape, which is defined as the fitness function evaluated on all points of the state space, first introduced by Wright (1932). Additionally, we investigate which factors affect the landscape, and consequently how the difficulty changes.

Understanding the behavior of the likelihood function for various mixture model data sets may provide insight into the complexity of the optimization of its landscape. The primary reason for why it is difficult to find the global optimum of Gaussian mixtures is that the likelihood function can have a large number of local optima that are quite inferior to the global optimum. The presence of many local optima on the fitness landscape increases the chance that optimization algorithms will get trapped in sub-optimal solutions.

In this chapter we look at previous work that characterize the difficulty of optimization problems. Törn et al. (1999) present three criteria that capture this difficulty. The authors do not provide a quantifiable metric for the third criterion, and so in this chapter we propose a new metric to quantify this criterion. Additionally, we propose a new, fourth criterion for measuring the difficulty of an optimization problem. We introduce a metric for measuring this fourth criterion, and examine the classes of problems in which this new criterion is deemed important. We apply these global landscape metrics to two classical optimization problems as well as a variety of Gaussian mixture model data sets, and show how different attributes of the data set affect these landscape measures.

The rest of the chapter begins with a discussion of metrics that quantify the difficulty of optimization problems in Section 5.2. We discuss how these metrics can be calculated and applied to Gaussian mixture models in Section 5.3. Then, in Section 5.4 we apply the metrics to several simulated and real-world data sets. We summarize the findings in the discussion in Section 5.5.

5.2 Measuring the Difficulty of Optimization Problems

Consider the following optimization problem:

$$\text{Min } f(y), \text{ s.t. } y \in \chi \subseteq \Re^n. \tag{5.1}$$

We assume that the optimization problem defined in Equation (5.1) has a unique global optimum y^{**}, and $\mathcal{N}$ locally optimal solutions, $y_1^*, y_2^*, ..., y_{\mathcal{N}}^*$, such that $f(y^{**}) < f(y_i^*)$, for all $i = 1, 2, ..., \mathcal{N}$.

Törn et al. (1999) discuss three criteria to consider when characterizing the difficulty of a global optimization problem. The first criterion, and arguably the most important, is the relative size of the *region of attraction* of the global optimum with respect to the size of the solution space. The region of attraction of a local optimum is similar to the idea of the stability region in nonlinear dynamical systems (Reddy et al., 2006). We define the region of attraction of a local optimum y_i^* of (5.1) as the subset of the solution space, $R(y_i^*) \subset \chi$, such that an infinitely small step, strictly decreasing local optimization algorithm starting in any point in $R(y_i^*)$ will converge to y_i^*. The difficulty of a given optimization problem is directly correlated to the relative size of the global optimum's region of attraction with the size of the solution space.

To further discuss mathematically what the region of attraction represents, we first define the operator $| \cdot |$ on a set. For a discrete set A, the value $|A|$ is simply defined as the number of elements in the set A. However, in this chapter we concentrate on continuous optimization problems, and so the spaces we consider are non-discrete, and generally compact. For a compact set $B \subset \Re^n$, we define $|B|$ as the Lebesgue measure of the set B.

We now define the relative size of the region of attraction of a local optimum y_i^*, given by $\mathcal{A}(y_i^*)$, as the ratio $\mathcal{A}(y_i^*) := |R(y_i^*)|/|\chi|$. Clearly, the value $\mathcal{A}(y_i^*)$ is a positive number between zero and one. If the region of attraction of the global optimum y^{**} is a relatively large portion of the solution space (i.e., $\mathcal{A}(y^{**})$ is close to one), then finding that global optimum is not difficult. Conversely, if $\mathcal{A}(y^{**})$ is relatively small (i.e., close to zero), then finding the global optimum is more difficult. However, we take careful note that for these purposes it is imperative to consider a solution space χ that is bounded. That is, we choose the bounds of χ in a way so that we are certain that the bounds contain only feasible solutions, and that χ contains the global optimum. We note that we can be certain that χ contains the global optimum for constrained formulations of optimization problems, such as the Gaussian mixture model formulation described in Chapter 4. In unconstrained optimization problems, finding proper bounds for χ may be more difficult. Determining the bounds of the solution space χ can drastically affect the value of the ratio $|R(y_i^*)|/|\chi|$. Therefore, we choose the bounds of χ in the same manner as we choose the bounds for random starting values in a solution space for multiple runs of a local optimization algorithm. It is of primary importance that the bounds are chosen in a consistent manner for the same class of problems to provide a fair comparison. We discuss how the bounds of χ are chosen for the parameters of Gaussian mixtures in Section 5.3.

The second criterion for the characterization of the difficulty of optimization problems is the number of unique local minima, $\mathcal{N}$. This idea is straightforward; the more local minima, the more opportunities for an optimization algorithm to get stuck in sub-optimal solutions. Furthermore, a large number of local minima corresponds to more computational time being spent using local search to investigate non-globally optimal local solutions.

The third and final criterion is the embeddedness of the global minimum. An embedded global optimum as one in which points sampled close to the global minimum will in general be better (in terms of fitness) than points sampled farther away from the global minimum. Similarly, local minima closer to the global minima are generally better solutions than local minima farther away from the global minimum. An isolated global optimum is a solution that is not embedded; that is, the corresponding optimization problem is more difficult to optimize because points in its surrounding neighborhood have relatively poor fitness compared to points farther away. Obviously an embedded global optimum would generally be easier to find on a fitness landscape than an isolated one, because evolutionary optimization methods steer in the direction of better fitness values in an attempt to obtain the global optimum. Törn et al. (1999) present this idea of embeddedness as a rather vague, high-level criterion for measuring the difficulty of an optimization problem. They do not discuss a quantifiable metric for gauging the embeddedness of a given problem. We present such a metric now.

Jones and Forrest (1995) present a fitness-distance correlation metric for optimization problems where the correlation is between the distances of *random points* in the solution space to the global optimum and the fitness values of these points. We introduce a metric that is similar, but instead we measure the correlation of the distances of the unique *locally optimal solutions* to the global optimum and their corresponding fitness values. So, for each locally optimal solution y_i^*, we compute the Euclidean distance, $\|y_i^* - y^{**}\|$. We then compute the correlation r of the (distance, fitness) pairs $(\|y_i^* - y^{**}\|, f(y_i^*))$, as given by Equation 5.2.

$$\begin{aligned} r &= \frac{\sum_{i=1}^{\mathcal{N}}(x_i - \bar{x})(z_i - \bar{z})}{(\mathcal{N}-1)s_x s_z}, \\ x_i &= \|y_i^* - y^{**}\|, \\ \bar{x} &= \frac{1}{\mathcal{N}}\sum_{i=1}^{\mathcal{N}} x_i, \\ s_x &= \sqrt{\frac{1}{\mathcal{N}-1}\sum_{i=1}^{\mathcal{N}}(x_i - \bar{x})^2}, \\ z_i &= f(y_i^*), \\ \bar{z} &= \frac{1}{\mathcal{N}}\sum_{i=1}^{\mathcal{N}} z_i, \\ s_z &= \sqrt{\frac{1}{\mathcal{N}-1}\sum_{i=1}^{\mathcal{N}}(z_i - \bar{z})^2}. \end{aligned} \tag{5.2}$$

A correlation value r close to 1 signifies that a minimization problem is highly embedded, while a value of r close to -1 for a minimization problem indicates an isolated global optimum, and vice versa for maximization problems.

5.2.1 A Fourth and New Attribute for Characterizing Optimization Problems

In the following, we propose a fourth and new criterion for measuring the difficulty of optimization problems. This criterion is the relative size of the region of attraction of ϵ-optimal solutions. The reasoning behind this criterion is that in certain problems a solution whose fitness value is within a small range ϵ of the global optimum may be considered good enough, especially when the marginal cost of finding the global optimum is significantly greater than the time required to produce an ϵ-optimal solution. We define the region of attraction of ϵ-optimal solutions as $R(y_\epsilon^{**}) = R(y^{**}) \cup \bigcup_{i=1}^{\mathcal{N}} \{R(y_i^*) \mid f(y_i^*) - f(y^{**}) < \epsilon\}$, and the corresponding relative size of the region of attraction of ϵ-optimal solutions as $\mathcal{A}(y_\epsilon^{**}) := \frac{|R(y_\epsilon^{**})|}{|\chi|}$.

For some optimization problems one may be satisfied with a solution whose objective value is within ϵ of the global optimum. For example, given a pool of financial assets, say we want to maximize the return given a certain amount of risk. Then, the focus is on the risk and expected return of the portfolio rather than the specific make-up of the portfolio. A second example is the traveling salesman problem, where we may be primarily interested in the distance traversed along the prescribed route, rather than the ordering of the sites visited. In both of these examples, one may be satisfied with a local solution that, while not optimal, has a fitness of within ϵ of the global optimum.

However, an ϵ-optimal solution that is significantly different from the global optimum may have adverse consequences in some problem settings. For example, in statistical applications that involve maximum likelihood estimation, the likelihood value itself usually has little physical meaning, and it is the optimized variables (e.g., parameters of a family of distributions) that are of primary importance, because they are used to make statistical inferences. Since an ϵ-optimal solution may be vastly different from the global optimum, the resulting statistical inferences are likely to differ drastically, as well.

5.3 Calculating the Global Landscape Metrics

We call the boundary of a region of attraction, or stability region, of a local optimum the stability boundary of that local optimum. Finding an analytical representation of the stability boundaries of a local optimum is extremely difficult, and sometimes impossible, for optimization problems with nonlinear objective functions. Furthermore, global optimization problems by nature tend to have nonlinear objective functions. For this reason an analytical computation of $|R(y_i^*)|$ is usually infeasible. Therefore, we estimate $\frac{|R(y_i^*)|}{|\chi|}$ empirically by randomly sampling over χ, applying a local descent algorithm to each of these random starting values, and calculating the percentage $\hat{p}$ of runs that converge to y_i^*. As noted earlier, we choose the bounds of χ in the same manner as we choose the bounds for random starting values in a solution space for multiple runs of a local optimization algorithm; that is, the bounds will contain the global optimum, and will only contain feasible solutions to the problem.

For the second criterion, finding the *exact* number of locally optimal solutions $\mathcal{N}$ for a nonlinear optimization problem may be infeasible. However, when emprirically estimating the relative size of the region of attraction of the global optimum described above, we can also estimate the number of locally optimal solutions by counting the unique number of locally optimal solutions found in the same experiment. Naturally, this number would only represent the lower bound of the unique number of local optima. But by increasing the number of random starting values used to an arbitrarily large value, we can obtain a reasonable estimate for the correct number. We estimate the embeddedness of the optimization problem by computing the correlation r of the unique locally optimal solutions found according to Equation (5.2).

5.3.1 Applying Global Landscape Metrics to Known Examples

We demonstrate the global landscape metrics discussed in Section 5.2 on two known examples. For both of the examples, we invoke a local optimization algorithm on 1000 starting values, uniformly distributed over the solution space. The local optimization algorithm we use is *fminsearch*, a built-in Matlab function that utilizes the simplex search method of Lagarias et al. (1998). We compute the relative size of the region of attraction of the global optimum $\mathcal{A}(y^{**})$, the number of unique locally optimal solutions $\mathcal{N}$, the correlation r of the (distance, fitness) pairs of the locally optimal solutions, and the relative size of the region of attraction of ϵ-optimal solutions $\mathcal{A}(y^{**}_{\epsilon})$. For the following experiments, we let $\epsilon = 1$, so that any solution with a fitness value within one unit of the global optimum is an ϵ-optimal solution.

5.3.1.1 Shekel's Foxholes

De Jong (1975) introduced the function known as Shekel's Foxholes, often known as function F5 in his test suite, which is now typically used as benchmarking for Genetic Algorithms. The function, given in Equation (5.3). has 24 local minima, excluding the unique global minimum at $f_1(-32, -32) \approx .998032$. Its landscape is notoriously difficult to optimize, and this is visually apparent as given by the plot of the landscape in Figure 5.1.

$$f_1(x_1, x_2) = \frac{1}{.002 + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ji})^6}}, \tag{5.3}$$

$$\begin{gathered} a_{j1} = \{32, 16, 0, 16, 32, 32, 16, 0, 16, 32, 32, 16, 0, 16, 32, \\ 32, 16, 0, 16, 32, 32, 16, 0, 16, 32\}, \\ a_{j2} = \{32, 32, 32, 32, 32, 16, 16, 16, 16, 16, 0, 0, 0, 0, 0, \\ 16, 16, 16, 16, 16, 32, 32, 32, 32, 32\}, \\ 65.536 \leq x_i \leq 65.536, \; i = 1, 2. \end{gathered}$$

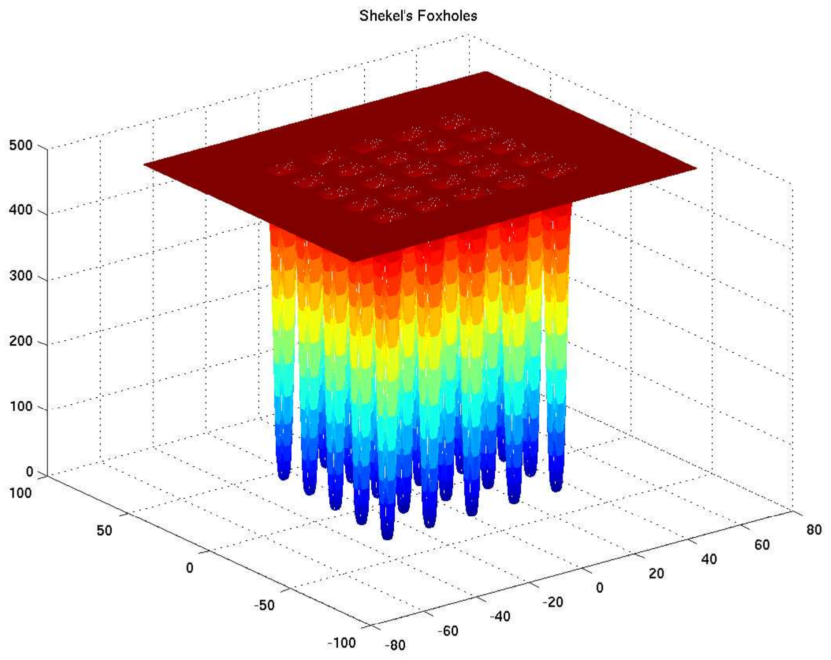

Figure 5.1: Plot of the landscape of Shekel's Foxholes.

The results of 1000 runs of *fminsearch* with random initializations are as follows. 51 of the 1000 runs converged to the global minimum, and therefore we estimate the region of attraction of the global minimum as 5.1%. Multiple runs converged to every of the $\mathcal{N} = 24$ unique local optima. One local minimum had a fitness value within $\epsilon = 1$ of the global minimum, which raises the number of ϵ-optimal runs to 101 out of 1000. Figure 5.2 shows the Euclidean distances of the local solutions plotted against their fitness values, along with the line of best fit. The correlation of the data in the plot is .8046, signifying a fairly highly embedded global optimum. The results of these runs are found in Table 5.1.

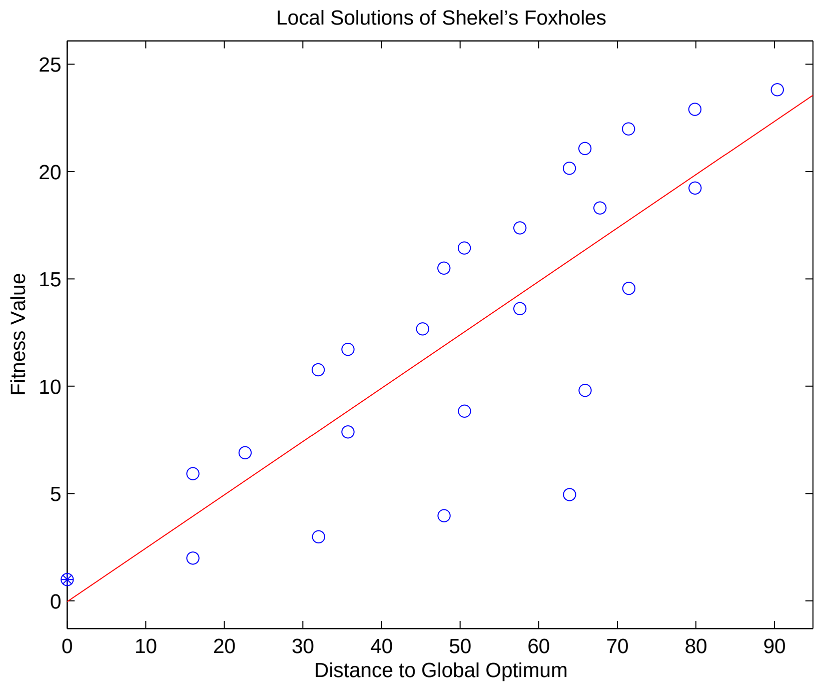

Figure 5.2: Plot of the local solutions of Shekel's Foxholes, with their distance to the global minimum plotted against their corresponding fitness values.

5.3.1.2 Goldstein-Price Function

The Goldstein-Price function is an eighth-degree polynomial in two variables, first introduced by Goldstein and Price (1971). The function, given in Equation 5.4, has a total of three local minima, excluding the global minimum at $f_2(0, -1) = 3$.

$$f_2(x_1,x_2) = \{1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3{x_2}^2)\}$$
$$\cdot \{30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12{x_1}^2 + 48x_2 - 36x_1x_2 + 27{x_2}^2)\}, \quad (5.4)$$
$$2 \le x_i \le 2, \ i = 1, 2.$$

The results of 1000 runs of *fminsearch* with random initializations are as follows. 598 of the 1000 runs converged to the global minimum, and therefore we estimate the relative size of the region of attraction of the global minimum as $\mathcal{A}(y^{**}) = 59.8\%$. The global minimum has a fitness value of 3, while the other three local minima have fitness values of 30, 84, and 840. Since no other local minimum is within $\epsilon = 1$ unit of the global minimum, then the region of attraction of ϵ-optimal solutions is equal to the region of attraction of the global minimum, so $\mathcal{A}(y_\epsilon^{**}) = 59.8\%$ as well. Multiple runs converged to all $\mathcal{N} = 3$ unique local optima. Figure 5.3 shows the Euclidean distances of the local solutions plotted against their fitness values, along with the line of best fit. The correlation of the (distance, fitness) pairs in the plot is $r = .6139$, indicating a moderately embedded global minimum. The results of these runs are found in Table 5.1.

Table 5.1: Simulation results of 1000 runs on Shekel's Foxholes and the Goldstein-Price function, with $\epsilon = 1$.

Function	$\mathcal{A}(y^{**})$	$\mathcal{N}$	r	$\mathcal{A}(y_\epsilon^{**})$
Shekel's	5.1%	24	.8046	10.1%
Goldstein-Price	59.8%	3	.6139	59.8%

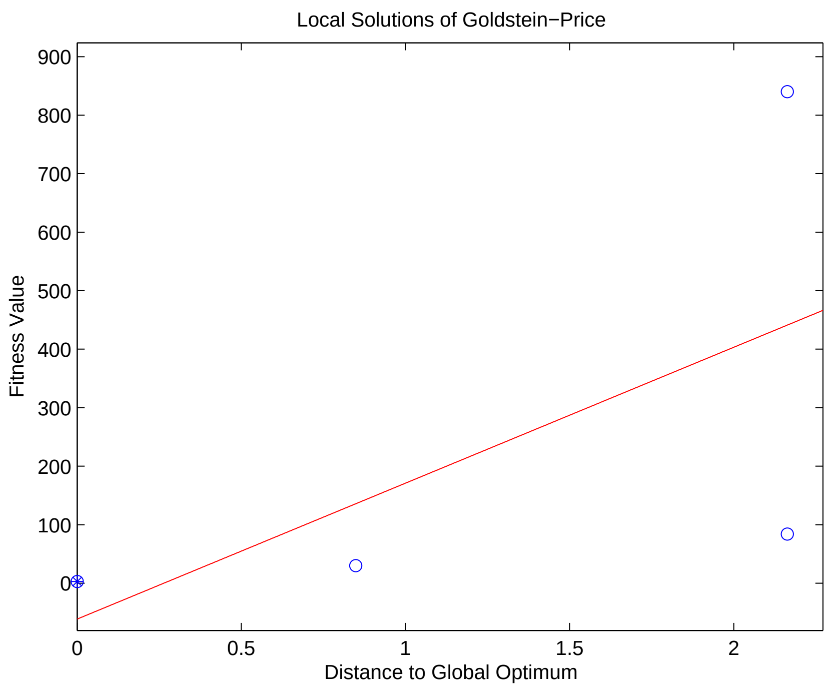

Figure 5.3: Plot of the local solutions of the Goldstein-Price function, with their distance to the global minimum plotted against their corresponding fitness values.

We notice several interesting results in Table 5.1. The primary reason that Shekel's Foxholes is a more difficult problem to optimize than the Goldstein-Price function is because the relative size of the global minimum of Shekel's Foxholes is significantly smaller than that of the Goldstein-Price function. Also, Shekel's Foxholes has 24 local optima compared to 3 local optima for the Goldstein-Price function, which means there are many more opportunities on the landscape of Shekel's Foxholes to get trapped in sub-optimal solutions. Shekel's Foxholes is more embedded than Goldstein-Price, which signifies that an evolutionary based optimization algorithm would have an easier time steering to the global minimum for Shekel's Foxholes. However, this may not be the case in reality, because the Goldstein-Price function has only three local minima, decreasing the statistical significance of its embeddedness correlation.

The data in Table 5.1 supports the fact that Shekel's Foxholes is a difficult problem to optimize. However, for Shekel's Foxholes, using a value of $\epsilon = 1$, the region of attraction of ϵ-optimal solutions is twice as large as the region of attraction of its global minimum. In other words, an optimization algorithm is twice as likely to obtain an ϵ-optimal solution to Shekel's Foxholes than obtaining the global minimum. Therefore, if one is willing to accept a solution relatively close to the global minimum, the chances of obtaining a *good enough* solution for this function are increased.

5.3.2 Applying Global Landscape Metrics to Gaussian Mixtures

Because of the nonlinearity of the likelihood function of Gaussian mixtures, finding an analytical closed-form expression for the region of attraction is not feasible. Therefore, we produce estimates for the relative size of the region of attraction, number of local optima, and the embeddedness correlation of the (distance, fitness) pairs of the local optima. We compute the distance between two clusterings simply as the Euclidean distance between their mean components, minimizing across the permuations of the labels. We do not consider the region of attraction of ϵ-optimal solutions, because, as addressed in Section 5.2.1, an ϵ-optimal solution may have far different optimal paramters than the global optimum of Gaussian mixtures. One particularly unique attribute of Gaussian mixtures is the presence of degenerate solutions, which we now address.

5.3.3 What to do with Degenerate Solutions?

Anytime we run EM from a random starting value, there is a possibility that the algorithm will converge to a degenerate solution. So, this raises two primary questions: what is the criterion for determining whether a solution is *degenerate*, and how should we incorporate degenerate solutions into the experimental results? We choose to answer the first question in the following way. For a given value of c, if a solution produced by EM violates the following constraint, we consider the solution to be degenerate (McLachlan and Peel, 2000):

$$\max_{i,j} \frac{|\Sigma_i|}{|\Sigma_j|} \leq c > 0. \tag{5.5}$$

For the simulated examples, we know the true global optimum and thus can choose a reasonable value of c intelligently. In particular, we choose the value of c in the experiments as follows. Since we know the true global optimum, we compute $c^* := \max_{i,j} \frac{|\Sigma_i|}{|\Sigma_j|}$ of the optimal solution. We then set the value of c equal to $25c^*$, rounding to the nearest 50. Therefore, we are assured to deem potentially promising clusterings as non-degenerate. However, in the case where the optimal solution is not known, it might be necessary to experiment with several values of c. The important thing is to use a value of c large enough so that the optimal solution is considered non-degenerate.

The second question is how we should incorporate the degenerate solutions in the results of the experiment. Since a degenerate solution can have an inflated likelihood value (since it's mean is centered on a single point, as the generalized variance of the covariance matrix tends to zero, the likelihood value tends to infinity), we choose not to include the solutions in the results. However, we count the number of degenerate solutions obtained in an experiment, becuase every degenerate solution obtained by EM is wasted computational time. Additionally, if a percentage p_{deg} of runs converge to degenerate solutons, then we consider p_{deg} as an estimate for the proportion of the solution space that is the degenerate region of attraction.

For the numerical experiments on Gaussian mixture models, we run So, if we apply EM to 1000 random starting values, we report the number of degenerate solutions obtained, and then the number of unique non-degenerate local optima obtained.

5.4 Numerical Examples

In the following, we perform the test on a number of different examples, including two simulated examples as well as two real-world examples. Throughout the examples, the solution space is considered to be as follows: the means between the minimum (min(X)) and maximum data points (max(X)), the weights between .1 and .9, and the variance components between .01 Var(X) and .5 Var(X). The lower bound for the variance components was chosen to be non-zero in an attempt to discourange small variance degenerate solutions, while the upper bound was chosen because all of the variance components of the optimal solutions of the data sets were below this bound. We initialize the covariance matrices as diagonal matrices, with the variance components along the diagonal. All initial values for EM were simulated from a uniform distribution across the solution space.

For each experiment, we apply EM to random starting values until we find the first 500 non-degenerate solutions. This way there will be a fair comparison for the number of local optima for each data set. However, we note that in these examples, unlike the known examples of Section 5.3.1, we do not know the true number of local optima, so we estimate $\mathcal{N}$ as a lower bound for the true value. In the following examples, we estimate the relative size of the region of attraction of the global optimum $\mathcal{A}(y^{**})$, the number of unique locally optimal solutions $\mathcal{N}$, the correlation r of the (distance, fitness) pairs of the locally optimal solutions, and the proportion of the solution space p_{deg} that is the degenerate region of attraction.

5.4.1 3-component Bivariate Gaussian Mixture

The first simulated example we analyze is the simple 3-component bivariate Gaussian mixture from Section 3.3.3. For reference, the graph of the optimal solution is given in Figure 5.4.

Applying EM to random starting values, we found the following results: using a degenerate score of $c = 150$, 80 runs converged to degenerate solutions before the 500^{th} non-degenerate solution. We estimate the relative size of the degenerate region of attraction as $p_{deg} = \frac{80}{580} = 13.79\%$. 448 of the 500 non-degenerate EM runs converged to the optimal value of -413.99. Thus we estimate the relative size of the global optimum's region of attraction to be $\mathcal{A}(y^{**}) = \frac{448}{580} = 77.24\%$ of the solution space. Of the non-degenerate runs, we found $\mathcal{N} = 14$ unique local optima. Figure 5.5 shows the plot of each of the non-degenerate local solutions found, with their

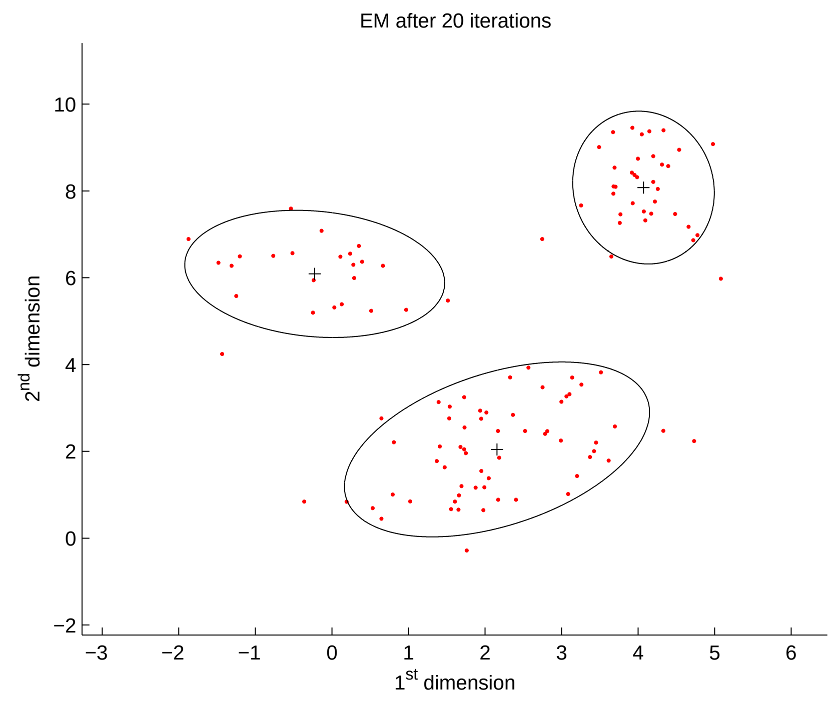

Figure 5.4: Plot of the simple bivariate Gaussian example with the optimal parameters.

distance to the global optimum plotted against their corresponding log-likelihood value. The data in Figure 5.5 has a correlation of $r = -0.7907$, signifying a fairly highly embedded global maximum. The results for the simulations on this data set, along with the other three data sets, are found in Table 5.2.

5.4.2 6-component Bivariate Gaussian Mixture

The second simulated example is the more difficult 6-component bivariate Gaussian mixture data set introduced in Section 3.3.3. The graph of the optimal clustering solution is given in Figure 5.6.

Applying EM to random starting values, we found the following results: using a degenerate score of $c = 1000$, 347 runs converged to degenerate solutions before

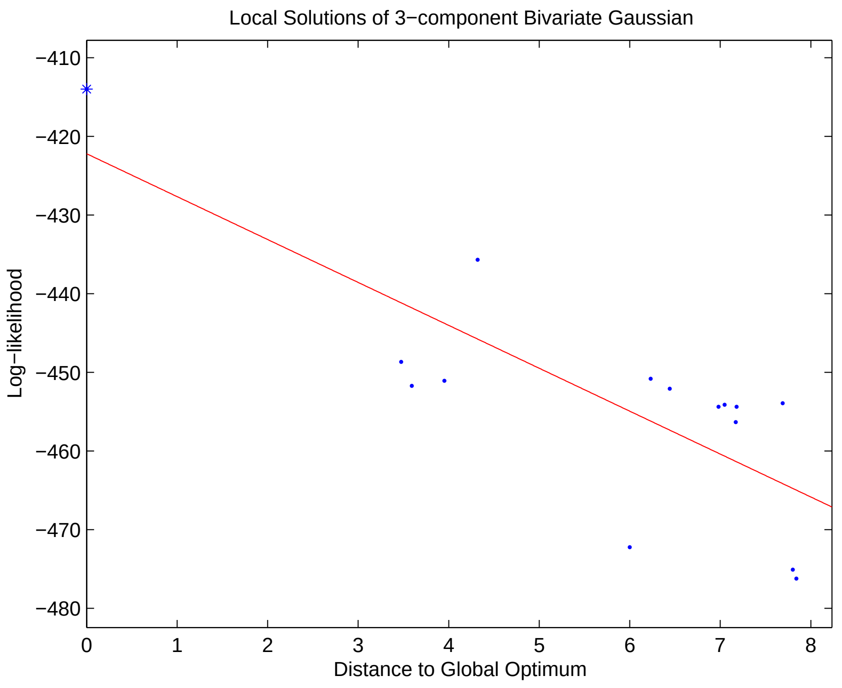

Figure 5.5: Plot of the local solutions of the simple bivariate Gaussian example, with their corresponding distance to the global optimum plotted against their corresponding log-likelihood values.

the 500th non-degenerate solution. We estimate the relative size of the degenerate region of attraction as $p_{deg} = \frac{347}{847} = 40.97\%$. 50 of the 500 non-degenerate EM runs converged to the optimal value of -946.98. Thus we estimate the relative size of the global optimum's region of attraction to be $\mathcal{A}(y^{**}) = \frac{50}{847} = 5.90\%$ of the solution space. Of the non-degenerate runs, we found $\mathcal{N} = 180$ unique local optima. Figure 5.7 shows the plot of each of the non-degenerate local solutions found, with their distance to the global optimum plotted against their corresponding log-likelihood value. The data in Figure 5.7 has a correlation of $r = -0.7872$, signifying a fairly highly embedded global maximum. The results for the simulations on this data set are found in Table 5.2, at the end of this section.

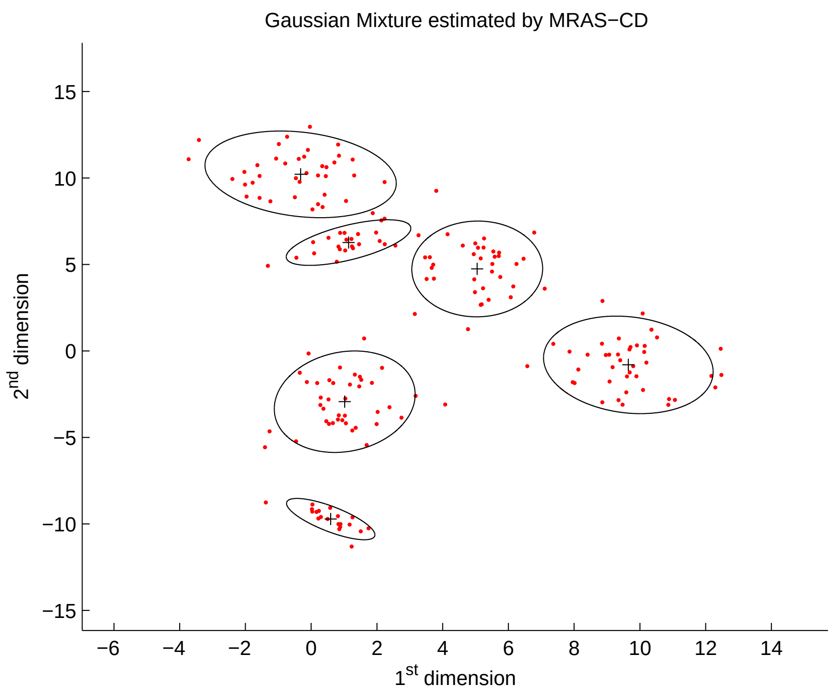

Figure 5.6: Plot of the more difficult bivariate Gaussian example with the optimal parameters.

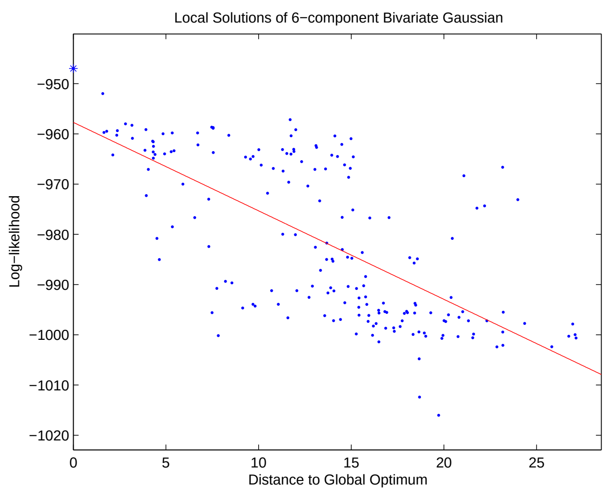

Figure 5.7: Plot of the local solutions of the more difficult bivariate Gaussian example, with their corresponding distance to the global optimum plotted against their corresponding log-likelihood values.

5.4.3 Iris Data Set

The iris data set consists of 150 random samples of iris flowers, including 50 samples from each of the following three iris species: setosa, versicolor, and virginica.

The data was collected by Anderson (1935). Each data sample consists of four measurements: sepal length, sepal width, petal length, and petal width, all measured in centimeters. The iris data set gained notoriety when Fisher (1936) developed a linear discriminant model to classify the species from the data measurements. Figure 5.8 depicts the 1st and 2nd principal components of the data, along with the optimal clustering solution, corresponding to a log-likelihood value of -180.19.

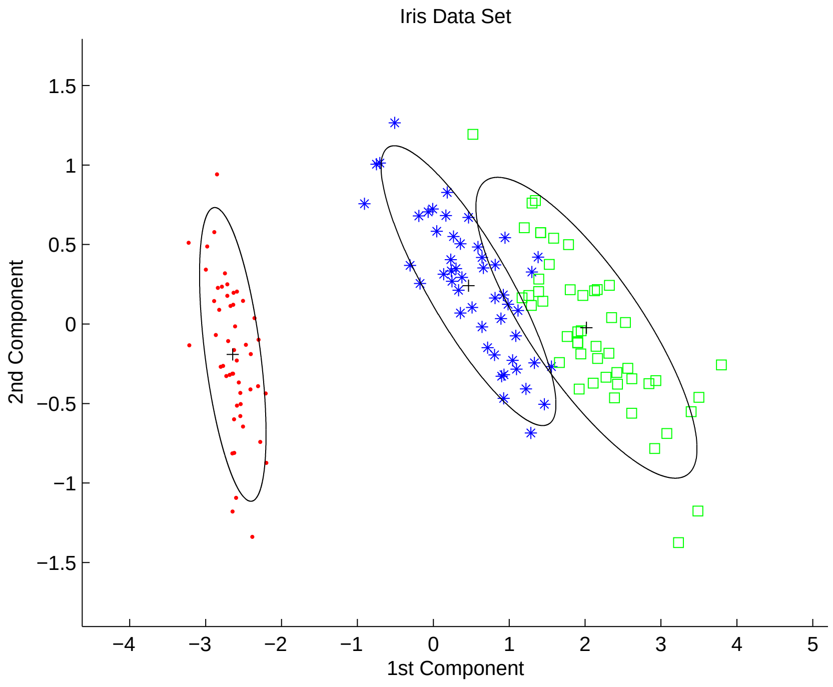

Figure 5.8: Plot of the 1st and 2nd principal components of the iris data set, along with the corresponding optimal solution.

Applying EM to random starting values, we found the following results: using a degenerate score of $c = 10,000$, 67 runs converged to degenerate solutions before the 500^{th} non-degenerate solution. We estimate the relative size of the degenerate region of attraction as $p_{deg} = \frac{67}{567} = 11.82\%$. 277 of the 500 non-degenerate EM runs converged to the optimal value of -180.19. Thus we estimate the relative size of the global optimum's region of attraction to be $\mathcal{A}(y^{**}) = \frac{277}{567} = 48.85\%$ of the solution space. Of the non-degenerate runs, we found $\mathcal{N} = 11$ unique local optima. Figure 5.9 shows the plot of each of the non-degenerate local solutions found, with their distance to the global optimum plotted against their corresponding log-likelihood value. The data in Figure 5.9 has a correlation of $r = -0.7362$, signifying a moderately embedded global maximum. The results for the simulations on this data set are found in Table 5.2, at the end of this section.

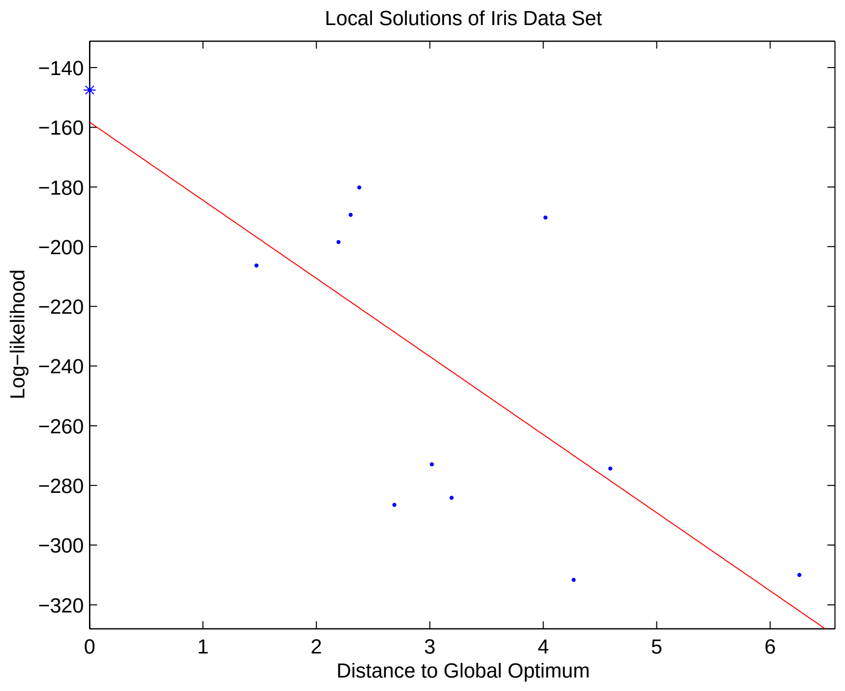

Figure 5.9: Plot of the local solutions of the iris data set, with their corresponding distance to the global optimum plotted against their corresponding log-likelihood values.

5.4.4 Control Chart Data

This data set consists of 600 control charts synthetically generated by the process in Alcock and Manolopoulos (1999). There are six different classes of control charts in the data: normal, cyclic, increasing trend, decreasing trend, upward shift, and downward shift. Each chart consists of a 60-dimensional vector. The goal is to cluster the charts into six groups, where each group represents one of the control chart classes. Figure 5.10 depicts the 1st and 2nd principal components of the data, along with the optimal clustering solution, corresponding to a log-likelihood value of -92799. Note that since we know the true memberships of each data point, in order to produce the global otpimum, we intiated each cluster component's parameters with that particular cluster's maximum likelihood paramters, and then ran EM.

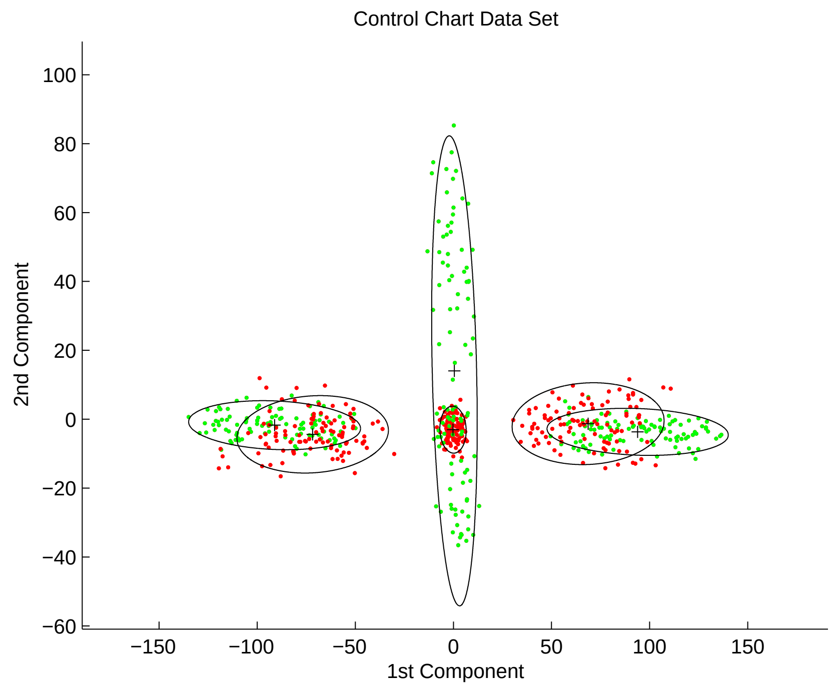

Figure 5.10: Plot of the 1st and 2nd principal components of the control chart data set, along with the corresponding optimal solution.

Applying EM to random starting values, we found the following results: using a degenerate score of $c = 10^{100}$, 521 runs converged to degenerate solutions before the 500^{th} non-degenerate solution. We estimate the relative size of the degenerate region of attraction as $p_{deg} = \frac{521}{1021} \approx 51.03\%$. Of the 500 non-degenerate runs, we found $\mathcal{N} = 439$ unique local optima. Perhaps due to the high dimensionality of the data, we found many local optima (a total of 429) with a better log-likelihood value than what we initially thought was the global optimum. The best solution found has a log-likelihood value of -72803. Because of the extremely high number of local optima, it is unclear whether this solution is the true global optimum of the data. Only a single run of the 500 non-degenerate solutions converged to the best solution found, and so we estimate the relative size of the global optimum's region of attraction to be $\mathcal{A}(y^{**}) = \frac{1}{1021} \approx 0.10\%$ of the solution space. Figure 5.11 shows the plot of each of the non-degenerate local solutions found, with their distance to the global optimum plotted against their corresponding log-likelihood value. The data in Figure 5.11 has a correlation of $r = -0.8317$, signifying a fairly highly embedded global maximum. The results for the simulations on this data set are found in Table 5.2, in the following section.

Figure 5.12 shows the plot of the 1st and 2nd principal components of many clusterings that have better log-likelihood values than the solution found in Figure 5.10. This is an interesting find, because if we did not know the true memberships of the data points, these results by clustering using Gaussian mixtures would lead us to believe that the true global optimum is a relatively poor solution, simply by comparing the log-likelihood values. In fact, two of the clusterings (the two on the right of Figure 5.12) lead us to believe that the data may only have 5 clusters, as 2 of the clusters sit on top of one another. This example shows us that there are instances where Gaussian mixture clusterings may have better likelihood values, yet

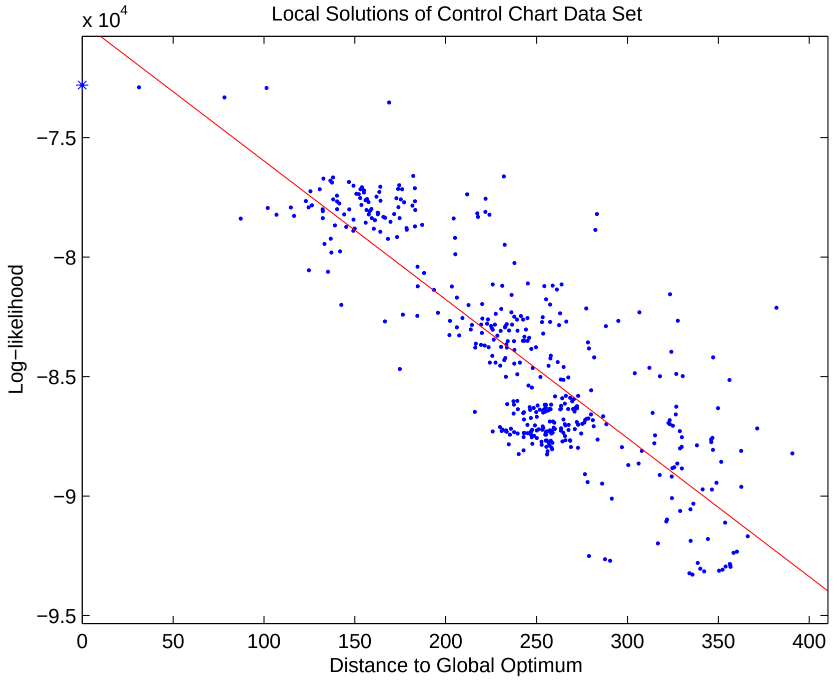

Figure 5.11: Plot of the local solutions of the control chart data set, with their corresponding distance to the global optimum plotted against their corresponding log-likelihood values.

in actuality do a poorer job of correctly classifying the true memberships of the data. As evidenced by this example, this occurrence may be due to the complexity brought on by the high dimensionality of the data.

5.4.5 Summary of Results

Table 5.2 summarizes the results of the four Gaussian mixture model examples. We report the number of clusters, dimensionality, relative size of the region of attraction of the global optimum, number of unique local optima found, embeddedness correlation, and the relative size of the degenerate region of attraction. Obviously the control chart data set is the most difficult to optimize, as evidenced by its low

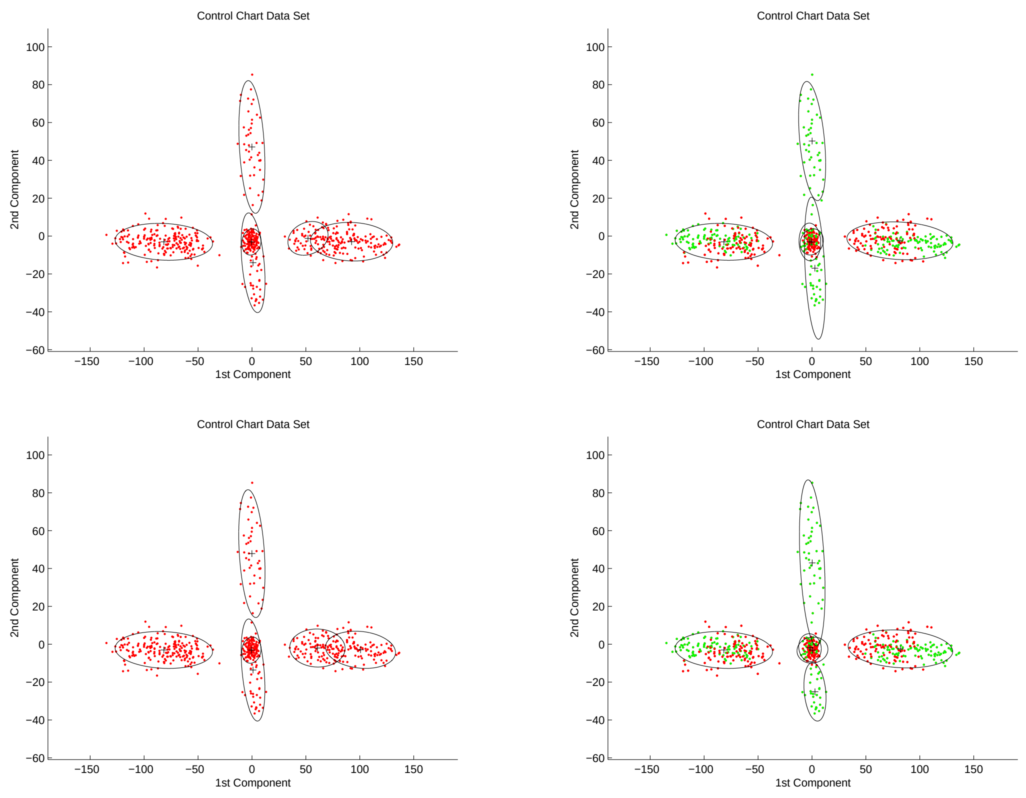

Figure 5.12: Several solutions of the clusterings of the control chart data which differ from the optimal solution, yet all have better log-likelihood values.

region of attraction of its global optimum score, as well as the high number of local optima found. In fact, the number of local optima for this data set may be orders of magnitude higher than the 439 reported, as these 439 were found in the first 500 non-degenerate runs. This may largely be in part to the high dimensionality of the control chart data set. The 6-component Gaussian mixture was the second most difficult to optimize. This leads us to believe that the number of clusters may be the key factor for difficulty in these problems. More clusters leads to more model parameters and thus more difficulty to find the global optimum. However, one attribute that we did not capture is how well separated are the clusters. This would also play an important role in characterizing the difficulty of optimizing Gaussian mixture models.

Table 5.2: Simulation results on four Gaussian mixtures.

Data Set	g	dimensionality	$\mathcal{A}(y^{**})$	$\mathcal{N}$	r	p_{deg}
3-component Gaussian	3	2	77.24%	14	-.7907	13.79%
6-component Gaussian	6	2	5.90%	180	-.7872	40.97%
Iris	3	4	48.85%	11	-.7362	11.82%
Control Chart	6	60	0.10%	439	-.8317	51.03%

We also notice that the embeddedness correlation coefficients were all fairly similar. This may signify that Gaussian mixture model problems are all roughly equivalent in terms of embeddedness, thus relegating the embeddedness criterion to be very minor in terms of its effect on optimization difficulty. Another interesting deduction from the table is that the problems with higher number of clusters also have a larger proportion of degenerate solutions. This could be attributed to the idea that the more clusters we optimize, the higher probability that at least one of them will get stuck in a degenerate solution centered on a single point.

5.5 Discussion

Many factors play a role in the difficulty of a given optimization problem. Some of these factors, such as the region of attraction of ϵ-optimal solutions, only have significance in certain problem settings. It would be nice to be able to gauge the difficulty of a given optimization problem with a single metric. However, as mentioned, this unified metric would likely be different for different classes of optimization problems, e.g., Gaussian mixture models being one. Also, in terms of mixture models, a metric for how well-separated are the clusters in a mixture would definitely help the characterization of the optimization difficulty. Furthermore, analyzing how the dimensionality, number of clusters, and number of data points of a data set affect these landscape measures would potentially lead to new and useful insight on the landscape of Gaussian mixtures.

Another extension to the work in this chapter could be to combine the results found for a given optimization problem with extreme value theory to estimate the true global optimum. Golden and Alt (1979) developed procedures for determining interval estimates for intractable global optima of NP-hard combinatorial optimization problems, and so an analogous procedure could be applied to mixture models.

Chapter 6

Conclusions

Perhaps the biggest current limitation of the CE and MRAS mixture model algorithms is the computational time to convergence. Because the proposed algorithms require the generation of multiple candidate solutions in each iteration, they are inherently more computationally-intensive algorithms than EM. This is similar in nature to the Monte Carlo EM algorithm (Booth and Hobert, 1999; Levine and Casella, 2001; Levine and Fan, 2003; Jank, 2004; Caffo et al., 2003), which spends most of its computational effort on simulating from a suitable distribution and is thus much slower than its deterministic counterpart. That being said, our current implementation of the CE and MRAS mixture model algorithms are not optimized for speed, and continuous advances in computing power and processor speed will make the computational disadvantages less practically important. At the end of the day, the decision that researchers faces is whether one wants fast but possibly highly inaccurate answers, or alternatively whether waiting a little longer is worth obtaining better solutions. The algorithms proposed in Chapter 3 are systematic ways for finding those solutions. However, part of my future plans include optimizing these algorithms for better run times. Additional future work would include generalizing the objective function in order to find the best clustering across all values of g. Also, we would like to characterize how the run-time complexity of the proposed algorithms changes with respect to problem size.

In Chapter 4 we presented a proof of global convergence of the MRAS-CD algorithm to the optimal solution for Gaussian mixtures. One possible path of future work could include extending the proof to finite mixture models of other probability distributions, in addition to Gaussian. In addition, proving the convergence of the other three algorithms presented in Chapter 3 is an open problem.

Chapter 5 discussed many factors that play a role in the difficulty of a given optimization problem. Some of these factors, such as the region of attraction of ϵ-optimal solutions, only have significance in certain problem settings. It would useful to combine all of the metrics discussed into a single metric. However, this unified metric would likely be different for different classes of optimization problems. Also, in terms of mixture models, a metric for how well-separated are the clusters in a mixture would characterize optimization difficulty. Furthermore, analyzing how the dimensionality, number of clusters, and number of data points of a data set affect these landscape measures would potentially lead to new and useful insights on the landscape of Gaussian mixtures.

Bibliography

Alcock, R. J. and Manolopoulos, Y. (1999). Time-Series Similarity Queries Employing a Feature-Based Approach. In *7th Hellenic Conference on Informatics*, Ioannina, Greece.

Anderson, E. (1935). The Irises of the Gaspé Peninsula. *Bulletin of the American Iris Society*, 59:2–5.

Biernacki, C., Celeux, G., and Govaert, G. (2003). Choosing Starting Values for the EM Algorithm for Getting the Highest Likelihood in Multivariate Gaussian Mixture Models. *Computational Statistics & Data Analysis*, 41(3-4):561–575.

Booth, J. G. and Hobert, J. P. (1999). Maximizing Generalized Linear Mixed Model Likelihoods with an Automated Monte Carlo EM Algorithm. *Journal of the Royal Statistical Society B*, 61:265–285.

Botev, Z. and Kroese, D. P. (2004). Global Likelihood Optimization via the Cross-Entropy Method with an Application to Mixture Models. In *Proceedings of the 2004 Winter Simulation Conference*.

Caffo, B. S., Jank, W. S., and Jones, G. L. (2003). Ascent-Based Monte Carlo EM. *Journal of the Royal Statistical Society B*, 67:235–252.

Cao, G. and West, M. (1996). Practical Bayesian Inference Using Mixtures of Mixtures. *Biometrics*, 52:1334–1341.

Celeux, G. and Govaert, G. (1992). A Classification EM Algorithm for Clustering and Two Stochastic Versions. *Computational Statistics & Data Analysis*, 14(3):315–332.

De Boer, P., Kroese, D. P., Mannor, S., and Rubinstein, R. Y. (2005). A Tutorial on the Cross-Entropy Method. *Annals of Operations Research*, 134:19–67.

De Jong, K. A. (1975). *An Analysis of the Behaviour of a Class of Genetic Adaptive System*. PhD thesis, University of Michigan, Ann Arbor, MI.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society B*, 39:1–22.

Diebolt, J. and Robert, C. P. (1990). Bayesian Estimation of Finite Mixture Distributions: Part II, Sampling Implementation. Technical Report III, Laboratoire de Statistique Théorique et Appliquée, Université Paris VI.

Fisher, R. A. (1936). The Use of Multiple Measurements in Taxonomic Problems. *Annals of Eugenics*, 7:179–188.

Fraley, C. and Raftery, A. E. (1998). How Many Clusters? Which Clustering Method? Answers via Model-Based Cluster Analysis. *The Computer Journal*, 41:578–588.

Golden, B. L. and Alt, F. B. (1979). Interval Estimation of a Global Optimum for Large Combinatorial Problems. *Naval Research Logistics Quarterly*, 26:69–77.

Goldstein, A. A. and Price, I. F. (1971). On Descent from Local Minima. *Mathematics of Computation*, 25(115):569–574.

Heath, J., Fu, M., and Jank, W. (2007a). Global Convergence of Model Reference Adaptive Search for Gaussian Mixtures. Working Paper, Smith School of Business, University of Maryland.

Heath, J., Fu, M., and Jank, W. (2007b). New Global Optimization Algorithms for Model-Based Clustering. Working Paper, Smith School of Business, University of Maryland.

Horn, R. A. and Johnson, C. R. (1990). *Matrix Analysis*. Cambridge Univ. Press, New York.

Hu, J., Fu, M. C., and Marcus, S. I. (2007). A Model Reference Adaptive Search Algorithm for Global Optimization. *Operations Research*, 55(3). In press.

Jamshidian, M. and Jennrich, R. I. (1993). Conjugate Gradient Acceleration of the EM Algorithm. *Journal of the American Statistical Association*, 88:221–228.

Jamshidian, M. and Jennrich, R. I. (1997). Acceleration of the EM Algorithm by using Quasi-Newton Methods. *Journal of the Royal Statistical Society B*, 59:569–587.

Jank, W. (2004). Quasi-Monte Carlo Sampling to Improve the Efficiency of Monte Carlo EM. *Computational Statistics & Data Analysis*, 48:685–701.

Jank, W. (2006a). Ascent EM for Fast and Global Model-Based Clustering: An Application to Curve-Clustering of Online Auctions. *Computational Statistics & Data Analysis*, 51(2):747–761.

Jank, W. (2006b). The EM Algorithm, Its Stochastic Implementation and Global Optimization: Some Challenges and Opportunities for OR. In Alt, F., Fu, M., and Golden, B., editors, *Perspectives in Operations Research: Papers in Honor of Saul Gass' 80th Birthday*, pages 367–392. Springer, New York.

Johnson, C. R. (1970). Positive Definite Matrices. *The American Mathematical Monthly*, 77(3):259–264.

Jones, T. and Forrest, S. (1995). Fitness Distance Correlation as a Measure of Problem Difficulty for Genetic Algorithms. In *Proceedings of the 6th International Conference on Genetic Algorithms*, pages 184–192.

Kroese, D. P., Porotsky, S., and Rubinstein, R. Y. (2006). The Cross-Entropy Method for Continuous Multi-Extremal Optimization. *Methodology and Computing in Applied Probability*, 8:383–407.

Kroese, D. P., Rubinstein, R. Y., and Taimre, T. (2007). Application of the Cross-Entropy Method to Clustering and Vector Quantization. *Journal of Global Optimization*, 37(1):137–157.

Lagarias, J. C., Reeds, J. A., Wright, M. H., and Wright, P. E. (1998). Convergence Properties of the Nelder-Mead Simplex Method in Low Dimensions. *SIAM Journal of Optimization*, 9(1):112–147.

Levine, R. and Casella, G. (2001). Implementations of the Monte Carlo EM Algorithm. *Journal of Computational and Graphical Statistics*, 10:422–439.

Levine, R. and Fan, J. (2003). An Automated (Markov Chain) Monte Carlo EM Algorithm. *Journal of Statistical Computation and Simulation*, 74:349–359.

Liu, C. and Rubin, D. B. (1994). The ECME algorithm: A Simple Extension of EM and ECM with Faster Monotone Convergence. *Biometrika*, 81:633–648.

Louis, T. A. (1982). Finding the Observed Information Matrix when Using the EM Algorithm. *Journal of the Royal Statistical Society B*, 44:226–233.

Mangiameli, P., Chen, S., and West, D. (1996). A Comparison of SOM Neural Network and Hierarchical Clustering Methods. *European Journal of Operational Research*, 93:402–417.

McLachlan, G. J. and Peel, D. (2000). *Finite Mixture Models.* Wiley, New York.

Meng, X. L. (1994). On the Rate of Convergence of the ECM Algorithm. *The Annals of Statistics*, 22:326–339.

Meng, X. L. and Rubin, D. B. (1993). Maximum Likelihood Estimation via the ECM Algorithm: A General Framework. *Biometrika*, 80:267–278.

Milligan, G. W. (1981). A Review of Monte Carlo Tests of Cluster Analysis. *Multivariate Behavioral Research*, 16:379–407.

Pinheiro, J. C. and Bates, D. M. (1996). Unconstrained Parameterizations for Variance-Covariance Matrices. *Statistics and Computing*, 6:289–296.

Reddy, C., Chiang, H., and Rajarathnam, B. (2006). Stability Region based Expectation Maximization for Model-Based Clustering. In *Proceedings of the IEEE/ACM International Conference on Data Mining (ICDM).*

Rubinstein, R. Y. (1997). Optimization of Computer Simulation Models with Rare Events. *European Journal of Operations Research*, 99:89–112.

Rubinstein, R. Y. (1999). The Simulated Entropy Method for Combinatorial and Continuous Optimization. *Methodology and Computing in Applied Probability*, 2:127–190.

Rubinstein, R. Y. and Kroese, D. P. (2004). *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation and Machine Learning*. Springer-Verlag, New York.

Thisted, R. A. (1988). *Elements of Statistical Computing*. Chapman & Hall, London.

Törn, A., Ali, M. M., and Viitanen, S. (1999). Stochastic Global Optimization: Problem Classes and Solution Techniques. *Journal of Global Optimization*, 14:437–447.

Tu, Y., Ball, M., and Jank, W. (2006). Estimating Flight Departure Delay Distributions - A Statistical Approach with Long-Term Trend and Short-Term Pattern. Forthcoming in the *Journal of the American Statistical Association*.

Ueda, N. and Nakano, R. (1998). Deterministic Annealing EM Algorithm. *Neural Networks*, 11(2):271–282.

Ward, Jr., J. H. (1963). Hierarchical Grouping to Optimize an Objective Function. *Journal of the American Statistical Association*, 58:236–244.

Wei, G. C. and Tanner, M. A. (1990). A Monte Carlo Implementation of the EM Algorithm and the Poor Man's Data Augmentation Algorithms. *Journal of the American Statistical Association*, 85:699–704.

Wright, S. (1932). The Roles of Mutation, Inbreeding, Crossbreeding, and Selection in Evolution. In *Proceedings of the Sixth International Congress on Genetics*, pages 355–366.

Wu, C. F. J. (1983). On the Convergence Properties of the EM Algorithm. *The Annals of Statistics*, 11(1):95–103.