Learning Probability Distributions

by

Sanjoy Dasgupta

B.A. Harvard University 1993

A dissertation submitted in partial satisfaction of the
requirements for the degree of
Doctor of Philosophy

in

Computer Science

in the

GRADUATE DIVISION
of the
UNIVERSITY of CALIFORNIA at BERKELEY

Committee in charge:

Professor Umesh V. Vazirani, Chair
Professor Peter J. Bickel
Professor Christos H. Papadimitriou

Spring 2000

Abstract

Learning Probability Distributions

by

Sanjoy Dasgupta
Doctor of Philosophy in Computer Science

University of California at Berkeley

Professor Umesh V. Vazirani, Chair

The first part of this thesis presents an algorithm which takes data from an unknown mixture of Gaussians in arbitrarily high dimension and recovers the parameters of this mixture to within the precision desired by the user. There are two restrictions on the mixture: its component Gaussians must be "well-separated" in a precise sense, and they must have a common (though of course unknown) non-singular covariance matrix. The running time of the algorithm is linear in the dimension of the data and polynomial in the number of Gaussians.

This algorithm is very simple, and relies crucially upon a particular procedure for dimensionality reduction. Data from a mixture of k Gaussians are projected to a randomly chosen $O(\log k)$-dimensional subspace, regardless of the original dimension and the number of data points, and it is shown that not only does this retain enough information for clustering, but it also causes a drastic reduction in the eccentricity of the Gaussians. Experiments are performed to illustrate some of the promise of this random projection technique.

The second half of this thesis studies the problem of learning the maximum-likelihood directed probabilistic net given data. Three combinatorial optimization tasks are distinguished: SL(k), learning the optimal probabilistic net in which each node has at most k parents; PT, learning the optimal polytree; and PT(k), learning the optimal polytree with an indegree bound of k. It is well-known that SL(1) is efficiently solvable while SL(2) is an NP-hard optimization problem. We demonstrate an even more damaging hardness barrier, that for some constant $c > 1$, unless P = NP there is no polynomial-time algorithm which can c-approximate SL(2), that is, which can consistently return a solution whose log-likelihood is within a multiplicative factor c of optimal.

We show a similar hardness result for PT(2), but at the same time prove that the optimal branching (or Chow-Liu tree), which can be found efficiently, constitutes a good approximation to the optimal polytree, regardless of indegree. The ratio between their log-likelihoods cannot be bounded by an universal constant but depends upon the nature of the individual attributes. For instance, if each attribute by itself has unit entropy then the ratio is at most two. An optimal structure over n nodes can have log-likelihood anywhere in the range $[-O(n), 0]$ and therefore this approximation result is a significant guarantee.

Professor Umesh V. Vazirani
Dissertation Committee Chair

Contents

Acknowledgements

I would like to thank my advisor, Umesh Vazirani, for guiding me with care through my time in graduate school, and providing practical advice and help on the many occasions when I needed them. My research also owes a lot to discussions with Yoav Freund, Nir Friedman, Anupam Gupta, Mike Jordan, Daphne Koller, Christos Papadimitriou, Stuart Russell, Leonard Schulman, Eric Vigoda, and David Zuckerman, and to the generosity and encouragement of Peter Bickel.

Chapter I

Introduction

The wide proliferation of computers has raised hopes that it might be possible to automatically analyze collections of data and provide concise, useful models of them. For a simple example of the kind of computational task involved, consider a gel electrophoresis experiment in which protein molecules in solution are mobilized by the application of a potential difference (Longsworth, 1942; Titterington, Smith and Makov, 1985). Speeds of some of these are then measured in the hope of gauging the relative proportions of different proteins. Molecules of a particular protein can be thought of as having an "ideal speed" μ which is perturbed by a host of environmental factors, like convection, of combined standard deviation some σ. The measured speed of one of these molecules can then conveniently be modeled as a Gaussian (normal) distribution with mean μ and variance σ^2, denoted $N(\mu, \sigma^2)$. This particular choice of distribution is prevalent throughout the physical sciences and has a universality that is captured in the central limit theorem (Feller, 1966).

If there are k distinct proteins in the solution, with ideal speeds $\mu_1, \mu_2, \ldots, \mu_k$ and noise levels $\sigma_1, \ldots, \sigma_k$, then the distribution of the measurements as a whole is a *mixture of Gaussians*,

$$w_1 N(\mu_1, \sigma_1^2) + w_2 N(\mu_2, \sigma_2^2) + \cdots + w_k N(\mu_k, \sigma_k^2),$$

where w_i is the proportion in which the i^{th} protein is present. The experimenter may decide that molecules of different proteins are subject to the same noise processes, in which case he can set $\sigma_1 = \sigma_2 = \cdots = \sigma_k$. The task then is to fit a mixture of Gaussians to the observations, as a useful summary of them. Once this is done, a potentially confusing list of measurements has been condensed into a brief characterization of each protein, which can be used to answer general questions like "is one protein much more common than the others?", and, if need be, to classify new measurements.

Can this kind of modelling be reliably computerized? What assurances could we reasonably expect? This issue is particularly important in the case of high-dimensional data, where it is difficult to visually check the computer's results. The need for performance guarantees in data analysis is the central preoccupation of this thesis.

1 Massive data sets

A natural way to model, or summarize, a data set is to think of it as consisting of samples from some unknown probability distribution, and to then learn a compact representation of this distribution. In a typical scenario, a learning algorithm is given a restricted family F of probability distributions and tries to pick one of them that best fits the data, according to some suitable criterion. The choice of family F is often based upon preconceptions about the data. In the electrophoresis example, a mixture of Gaussians was appropriate because it was anticipated that the data would form several clusters, each of which could be approximately described as normal. The goodness-of-fit criterion must also be specified, and will depend on the family of distributions, on additional prior expectations about the data, and on the specific uses to which the learned model will be put.

It is reasonable to select a family F that is believed to contain distributions close to the empirical distribution of the data, but this is not necessary. Even a modest or poor fit to the data might provide some valuable information. This is an important consideration in handling collections of information which are so vast as to preclude any direct human appraisal. Such collections, called "massive data sets" by a somewhat apprehensive scientific community, are now widespread (Kettenring and Pregibon, eds., 1996). They are being created continuously, spooled onto magnetic media from weather reports, credit card transactions, medical histories, handwriting samples, gene sequences, . . . a flood of data, accumulating day and night and awaiting attention.

One such electronic archive which has been studied in detail consists of articles from the Reuters newsfeed (Lewis, 1996). A common preprocessing step identifies $n = 10,000$ or more key words, counts the occurrences of these words in all the documents, and then represents each document as the corresponding n-dimensional vector of frequencies. Under this interpretation the archive consists of a vast number of data points, one per article, in the very high-dimensional space $\mathbb{R}^n$. A possible goal is to group together similar documents in the hope that this will facilitate subsequent queries.

The impressive number of samples in such a database poses various computational difficulties. It brings up issues of caching and paging, and strongly discourages too many passes through the data. However the more serious and fundamental concern is the high number of attributes per data point, which has been nicknamed "the curse of dimensionality" (Bellman, 1961).

What exactly is this curse? First, high-dimensional data is very hard to visualize. Finding the mean and covariance, or looking at various two-dimensional projections, can be helpful but also misleading. Diaconis and Freedman (1984) have demonstrated simple families of non-Gaussian distributions most of whose two-dimensional projections look Gaussian. Second, high-dimensional geometry contains many counter-intuitive pitfalls. One such effect can be observed by choosing points uniformly at random in an n-sphere and noticing that they almost always lie close to the surface of the sphere. We shall encounter quite a few more in the coming chapters. These two problems partially explain the basic difficulty, which is this: much of current algorithmic technology is very clearly directed toward low-dimensional data and becomes impractical when naively scaled to higher dimension. For instance, Voronoi diagrams are a staple of two- and three-dimensional computational geometry but lead to exponentially complex partitions of space when the dimension is increased; m points in $\mathbb{R}^n$ will typically generate Voronoi cells with $\Omega(m^n)$ faces.

2 Learning probability distributions

This thesis studies in detail two popular families of probability distributions. They are most easily introduced by example.

The Infrared Astronomical Satellite (IRAS) data set, described by Cheeseman *et al.* (1995), contains 5425 points in 100-dimensional space, each representing information from one of an unknown number of distinct galaxies. The authors decided to model the data from each galaxy by a high-dimensional Gaussian. These distributions are natural generalizations of the univariate Gaussian presented earlier. A Gaussian in n dimensions is indexed by its mean $\mu \in \mathbb{R}^n$ and covariance matrix Σ, and is denoted $N(\mu, \Sigma)$. It assigns to a point x in space the density

$$p(x) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right),$$

where $|\Sigma|$ is shorthand for the determinant of Σ. If Σ is diagonal, then points from the Gaussian have independent coordinates. The researchers analyzing the IRAS data made this choice, presumably to reduce the number of parameters involved. Their learning task was then to determine the number of clusters k and to fit a mixture of k Gaussians to the data. They found 77 clusters, and their results revealed some features previously unknown to astronomers.

The mixture of Gaussians is among the most enduring, well-weathered models of applied statistics. Its fundamental importance has made it the object of close experimental and theoretical study for well over a century. In a typical application, as in the two we have seen, sample data are thought of as arising from various possible sources, each present in a certain proportion called its *mixing weight*, and each modeled as a Gaussian. Given unlabeled data from these sources, the goal is to identify the mixture of Gaussians which generated them. The most difficult scenario is that in which there is no prior information either about the number of sources or about their nature; all this must be gleaned from the data.

For some data sets Gaussians are inappropriate models of clusters, and mixtures of more suitable distributions are used. For instance, if the data are all in $\{0,1\}^n$, it might be a better idea to model each cluster by, say, a product of Bernoulli (that is, $\{0,1\}$-restricted) distributions than by a Gaussian with diagonal covariance matrix. Mixtures as a whole are natural and convenient probabilistic models of clustered data. We will consider them in a unified framework after first treating the canonical case, that of mixtures of Gaussians.

A Gaussian distribution over $(X_1, \ldots, X_n)$ is specified by the individual means $\mathbf{E}X_i$ and by the second-order moments $\mathbf{E}X_iX_j$. But sometimes data contains more complicated relationships between the variables. For instance, it may be the case that X_1 is a function of X_2 and X_3. When there are many such strong dependencies which are not captured by pairwise correlations, it is often useful to model the data by a *directed probabilistic net*. As a toy example, consider the three random variables `Fever`, `Headache`, and `SoreThroat`, which can take on values `yes` or `no` and are defined on a sample space of some group of individuals. We would certainly expect strong correlations between these variables. Say we draw some data which makes it seem that `Headache` and `SoreThroat` are almost independent of each

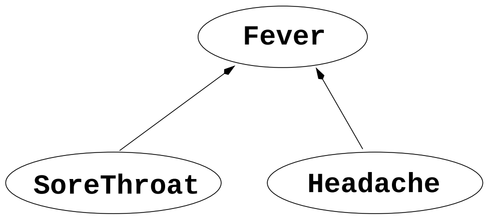


Figure 1: A toy probabilistic net.

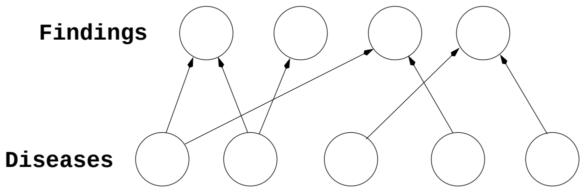


Figure 2: The QMR-DT probabilistic net has a bipartite structure.

other, but that they both make `Fever` more likely. This can be represented by the graph in Figure 1 or equivalently by the factorization

$$\begin{aligned} \mathbf{P}(\texttt{Headache}, \texttt{SoreThroat}, \texttt{Fever}) &= \mathbf{P}(\texttt{Headache}) \times \mathbf{P}(\texttt{SoreThroat}) \times \\ &\quad \mathbf{P}(\texttt{Fever} \mid \texttt{Headache}, \texttt{SoreThroat}). \end{aligned}$$

A more sophisticated example is the QMR-DT medical network (Shwe *et al.*, 1989; Jaakkola and Jordan, 1999), a graphical model which attempts to encapsulate medical knowledge about certain diseases and findings (symptoms). It contains nodes for approximately 600 diseases and 4,000 findings. These are arranged into a bipartite graph, with edges from diseases to findings (Figure 2); the common jargon for this is that the disease nodes are *sources* of the graph, and are *parents* of the symptom nodes. The network is annotated with the conditional probability distribution of each node given its parents, and these then specify a joint probability distribution over all the variables. A particularly appealing feature of such networks is that they are easy to read and immediately communicate some basic correlations between the different variables. The network structure can also in some cases be constructed, or constrained, by consultation with experts. In such cases, as in the QMR-DT net, it is common for the edges to reflect some interpretation of causality among the variables.

Directed probabilistic nets are always acyclic. A maximally connected acyclic net can

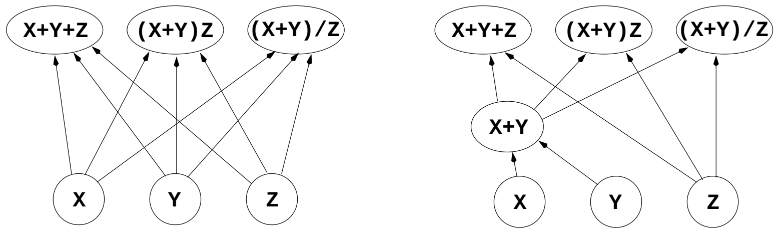


Figure 3: Simplifying a model by introducing a new variable.

represent any distribution over the variables and requires potentially enormous conditional probability distributions to be stored. In constructing these networks, therefore, a primary concern is to limit connectivity, or more specifically, to limit the number of parents of each node.

Probabilistic nets have become popular because they permit some complicated probability distributions to be expressed in a compact and easily understandable manner. Sometimes *hidden nodes* are introduced, representing random variables which are not directly observed but which either permit a more concise model or correspond to some hypothesized underlying causal mechanism. Figure 3, for instance, shows how a graph on six observable variables can be simplified by the creation of a hidden node.

Once a probabilistic net has been created, it can be used to address conditional probability queries about the distribution. For example, given the QMR-DT network, one might reasonably ask, "what is the chance that I have so-and-so disease if I am exhibiting these symptoms?" Answering such questions is called *inference* and in general is NP-hard to perform with even approximate accuracy (Cooper, 1990; Dagum and Luby, 1993). There are certain classes of networks for which efficient exact inference is possible, however. The most natural such class is that of directed trees, called *polytrees* in the artificial intelligence literature (Pearl, 1988).

It is becoming increasingly common to construct complex probability models out of simpler ones. For instance, many speech recognition systems use both mixture models and probabilistic nets as subcomponents. Depending upon style of speaking and environmental factors, there are many different sequences of sound signals which can convey the same word, or syllable, or phoneme. Therefore, for any particular phoneme, there is some distribution over these various possible acoustic sequences. It is standard to summarize this distribution by a *hidden Markov model* (Rabiner, 1989; Jelinek, 1997). Figure 4 is an example of such

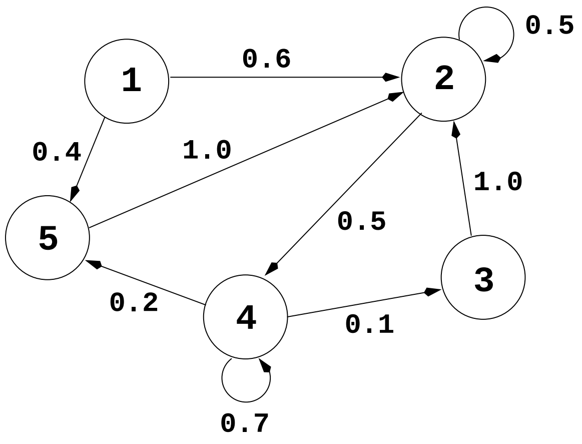


Figure 4: The underlying state graph of a simple hidden Markov model. The numbers on the edges are transition probabilities.

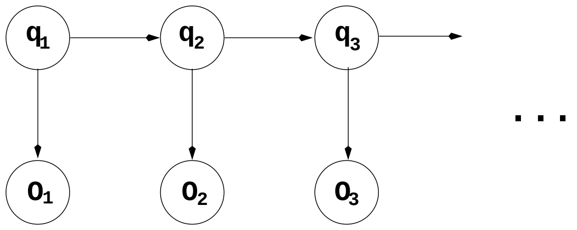


Figure 5: A time-slice depiction of the Markov model of Figure 4. Here q_t is the state (1-5) at time t, and O_t is the observation at time t.

a model. The graph is annotated by an initial distribution over the five states (nodes) and by transition probabilities on the edges. In addition, each state has its own distribution over sound signals, called its observable distribution. A sound sequence can be generated by performing a standard random walk on this graph, and at each time step choosing a sound from the observable distribution of the current state. The graph is sometimes drawn as a directed probabilistic net with nodes for the state and observation at each time step (Figure 5). The observable distribution of each state is typically a mixture of Gaussians. Sometimes, different states share the same set of Gaussians and are allowed to vary only the mixing weights ("tied mixtures").

For our purposes, the task of *learning* a mixture of Gaussians given a data set consists of finding the parameters of a mixture which fits the data well. In order to assess the quality of this fit, we will consider a simplified situation in which the data is actually generated from a mixture of Gaussians, which we call the "true" mixture, and the number of Gaussians is

The user provides:

- ϵ, an accuracy parameter;
- δ, a confidence parameter;
- k, the true number of Gaussians;
- w_{min}, a lower bound on the true mixing weights of the Gaussians; and finally,
- a source of i.i.d. samples from the true mixture.

A learning algorithm must now return a mixture of k Gaussians, such that with probability $> 1 - \delta$, the L_2 distance between each learned center and the corresponding true center is at most ϵ times the *radius* of that true Gaussian (defined in the next chapter). A learning algorithm is considered *efficient* if it runs in time polynomial in k and $1/\epsilon$, and polynomial in the dimension of the data.

Figure 6: The mixture of Gaussians learning framework.

known. The learned mixture is then evaluated on the basis of its proximity to the truth, specifically the Euclidean (L_2) distance between the learned Gaussian centers and the true centers. This learning scenario, summarized in Figure 6, is inspired by Valiant's (1984) "probably approximately correct" (PAC) model.

Our learning model for directed probabilistic nets is rather different. It is framed as a combinatorial optimization problem: given data, find the model with highest *likelihood*, that is, the model which assigns the highest probability to the data set. In this case, there are no assumptions whatever about the data and no notion of a "true" model. An efficient learning algorithm can then routinely be defined as one whose time and sample complexity are polynomial in the number of nodes.

It is always easy to find a good directed probabilistic net which has high connectivity, but this defeats the purpose of such models. A natural restriction on the complexity of a net is a bound on the number of parents its nodes are allowed to have. This motivates the triplet of optimization problems presented in Figure 7. In each case, there are no hidden nodes, so the number of nodes in the graph equals the number of attributes of the data set.

SL(k)	Find a directed probabilistic net of minimal cost in which each node has at most k parents.
PT	Find a polytree of minimal cost.
PT(k)	Find a polytree of minimal cost in which each node has at most k parents.

Figure 7: Learning directed probabilistic nets: some optimization problems. In each case, the cost of a model is its negative log-likelihood.

The cost of any given model is defined as its *negative log-likelihood* (that is, minus the log of its likelihood).

3 Current practice

The most popular technique for learning a mixture of Gaussians from data is currently the expectation-maximization (EM) algorithm. This is a local search heuristic of appealing simplicity, whose principal goal is convergence to a local maximum in the space of Gaussian mixtures ranked by likelihood. No performance guarantees of the kind outlined above are known for this algorithm.

For learning probabilistic nets, a variety of different local search algorithms have been proposed, including variants of gradient descent and EM. One important subcase stands out, that of *branchings* (sometimes called *Chow-Liu trees* in the literature), directed nets in which each node has at most one parent. A simple and computationally efficient algorithm for determining the maximum-likelihood branching (in the absence of hidden nodes) was discovered by Edmonds in 1967. In our terminology, this means that SL(1) (or equivalently, PT(1)) is efficiently solvable. To date this remains the central positive result in the literature on learning the structure of probabilistic nets.

The research that will be presented here is motivated principally by the lack of simple and provably good algorithms for learning important classes of probability distributions. As in the two learning frameworks just described, we shall think of a provably good algorithm as one which is computationally efficient and is accompanied by strong guarantees on the quality of fit of the learned model. Such guarantees give the user some indication of how much confidence he can have in the final answer and provide a scale along which different algorithms can be meaningfully compared. They are crucial in the construction of complex probabilistic models which rely upon the correct functioning of each component submodel.

4 A summary of results

Chapters II and III present an efficient learning algorithm, in the sense of Figure 6, for mixtures of Gaussians in arbitrarily high dimension. There are two restrictions: the component Gaussians of the true mixture must be *well-separated* (this notion is made precise in the next chapter), and they must have a common (though unknown) non-singular covariance matrix. The running time is linear in the dimension of the data and polynomial in the number of Gaussians.

This algorithm is very simple and relies crucially upon a particular procedure for dimensionality reduction. Data from a mixture of k Gaussians are projected to a randomly chosen $O(\log k)$-dimensional subspace, regardless of the original dimension and the number of data points, and it is shown that not only does this retain enough information for clustering, but it also causes a drastic reduction in the *eccentricity* of the Gaussians. This latter quantity will be defined later; for the time being, it should be interpreted as some natural measure of how non-spherical a Gaussian looks. Experiments in Chapter II illustrate some of the promise of this random projection technique.

Can something similar be done when the clusters do not look Gaussian, for instance if the data is discrete-valued? A celebrated result of Diaconis and Freedman (1984) demonstrates that many families of distributions, including for instance product distributions on the hypercube, look more Gaussian when randomly projected to low dimension, in a sense that we will make precise in Chapter III. Thus the techniques we develop might work for a fairly broad class of mixture models.

The last two chapters study probabilistic nets in terms of SL and PT. These are NP-hard optimization problems and therefore the main question is whether good approximation algorithms exist for them. A c-approximation algorithm for a minimization problem is defined as one which always returns a solution whose cost is within a multiplicative factor c of optimal. We have already seen that SL(1) is efficiently solvable. Chapters IV and V will demonstrate that SL(2) and PT(2) are, however, hard to even approximately solve. Specifically, there is some constant $c > 1$ for which c-approximating SL(2) is an NP-hard optimization problem. A similar result holds for PT(2).

The main result of Chapter V is an approximation algorithm for PT. It is shown that the optimal solution to PT(1), which can be obtained efficiently, constitutes a good approximation to the optimal polytree, regardless of degree. The multiplicative factor involved is not a universal constant but depends upon the nature of the individual attributes. For instance, if each attribute by itself has the distribution of a fair coin flip, then the factor is two. An optimal structure over n nodes can have cost anywhere in the range $[0, O(n)]$ and therefore this approximation result is a very significant guarantee.

Chapter II

Mixtures of Gaussians

1 Assessing the quality of a fit

In the maximum-likelihood (ML) framework, given data $S = \{x_1, \ldots, x_m\}$ the goal is to find a mixture of Gaussians θ, drawn from some family Θ (for instance, all mixtures of three Gaussians), which maximizes

$$\prod_{i=1}^{m} \theta(x_i),$$

$\theta(x)$ being the density of the mixture θ at point x. In other words, the goal is to maximize the probability density of the data S with respect to θ, and there are no assumptions about the data – in particular, it is not assumed that the data actually come from a mixture of Gaussians.

This framework has an appealing generality. A formal justification would ideally include a finite-sample guarantee of the form: "for any mixture θ, for any $\epsilon > 0$, there exists an m_ϵ such that the maximum-likelihood mixture of Gaussians with respect to m_ϵ i.i.d. samples from θ will with high probability have centers which are ϵ-close to those of θ, under some natural measure of closeness". Unfortunately, this is quite dramatically false. Given any data points $x_1, \ldots, x_m$, consider a mixture $\widehat{\theta}$ which centers one Gaussian on x_1 and allows its variance (or covariance matrix) to go to zero. Then $\widehat{\theta}(x_1)$ approaches infinity. In short, given any finite amount of data, the ML solution is ill-defined because a mixture with arbitrarily high likelihood can be found.

Define the *radius* of a Gaussian $N(\mu, \Sigma)$ to be $\sqrt{\text{trace}(\Sigma)}$ – we will soon see the reasons for this choice. Suppose that during ML learning, some lower bound is placed on the radius of the Gaussians in the mixture. Then the singularities we just encountered will no longer occur to quite the same alarming extent. Nevertheless, similar difficulties will continue to be present, since the creation of one (or more) clusters of very small radius can artificially boost the overall likelihood. It is perhaps fair to say that the ML framework is inadequate in situations where different clusters may have widely differing radii. However, this should not be judged too harshly since it is an important open problem in clustering to develop goodness-of-fit criteria that can satisfactorily handle such cases. The algorithms presented in this thesis all assume clusters of approximately the same radius.

Open Problem 1 Say m i.i.d. data points are generated from an unknown mixture θ of k Gaussians with the same covariance matrix, and then the maximum-likelihood solution $\widehat{\theta}$ is found within this same class of mixtures. Can it be shown that with high probability the Gaussian centers of $\widehat{\theta}$ will be close in L_2 distance to those of θ? What bounds can be given for this error as a function of k and m?

How can the maximum-likelihood solution be found? No algorithm is known which affords meaningful finite-sample guarantees about the quality of its solution vis-à-vis the optimal solution, even for very restricted hypothesis classes like mixtures of two spherical Gaussians. In practice, the likelihood space of Gaussian mixtures is explored using a local search technique like EM, which aims to find a local maximum. The quality of this solution then depends upon the prevalence and positioning of local maxima within the search space. Analyzing the performance of such algorithms, even with heavy assumptions about the data, is a challenging and fascinating direction of research.

Open Problem 2 What kind of guarantees can be given for EM's performance on finite data sets drawn i.i.d. from a mixture of Gaussians?

The next chapter describes an algorithm which takes data from a mixture of Gaussians and then recovers that mixture with high probability, to any specified accuracy. Why is it at all acceptable to make such a strong distributional assumption about the data? First, any algorithm with such a performance guarantee must be able to handle a certain amount of sampling error; thus it can automatically handle component distributions which are close to Gaussian in a specific sense that we will discuss in Chapter III. Second, the Gaussian is a distribution which occurs naturally in many data sets, for instance as noise in physical data. This phenomenon can be explained, after the fact, by appeal to entropy considerations or to the central limit theorem. Third, we will see that many non-Gaussian distributions can be made to look more Gaussian under a simple form of projection; this is related to the previous point and gives the Gaussian a certain universality.

For many clustering problems, it is impossible to give algorithms with useful performance guarantees under completely arbitrary conditions. Consider, for instance, the k-center problem.

> k-CENTER
>
> Input: n data points in $\mathbb{R}^d$; integer k.
>
> Output: k "centers" $\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}^d$ and radius r such that the data set is contained within $B(\mu_1; r) \cup \cdots \cup B(\mu_k; r)$.
>
> Objective: Minimize r.

It has been shown that there is no polynomial-time algorithm which can consistently find an r within a multiplicative factor two of optimal, unless P = NP, and that this factor can be achieved by a simple greedy algorithm (Hochbaum and Shmoys, 1985; González, 1985; Feder and Greene, 1988). This is a very weak guarantee which can be quite unacceptable for clustering. For instance, suppose each cluster in the data looks like the surface of a sphere. Then choosing "centers" on the surfaces of these spheres will yield a value of r within a factor two of optimal, and yet it should be possible to do much better.

A big challenge in designing an algorithm for a combinatorial optimization problem is to state the problem in an useful manner. If it is stated in too much generality, then it might be impossible to concoct an algorithm with any meaningful performance guarantee. In this case, there may be many algorithms to choose between, but one will not be able to differentiate between them – the goodness criterion is so stringent that they all fail miserably. One could run simulations or make a few heuristic arguments for specific cases, but overall it will be hard to make a definitive and satisfactory choice. This may well be the case with the ML formulation of the mixture-of-Gaussians problem without any assumptions on the data.

Clustering is a hard problem which is the subject of intensive research. One possible mode of attack is to alternate exploratory data analysis with algorithm design:

> Explore data sets of interest to gain some insights into them. What do clusters look like?
>
> Using these insights, formalize some assumptions about the data. Design an algorithm which works well under these assumptions.

A reasonable way of starting this process is by designing a clustering algorithm which works under what is perhaps the most widely used assumption – that the data look like a mixture of Gaussians.

2 Learning algorithms

The reference book of Titterington, Smith and Makov (1985) contains a brief history of the many fascinating and idiosyncratic techniques that have been applied to learning mixtures of Gaussians. These range from manual univariate procedures requiring graph paper and sharpened pencils to more sophisticated numerical methods based upon the manipulation of empirical Fourier coefficients of the data. Further details and more recent advances can be found in an excellent survey article by Redner and Walker (1984) and in a monograph by Lindsay (1995). Among the various algorithms, two of the most prominent are *the method of moments* and EM.

2.1 The method of moments

The theory of mixture models is often thought to have started with Pearson's (1894) clustering of crabs. The data were collected from 1000 crabs in Naples. For each crab, the ratio of the "forehead" to the body length was recorded. Pearson wanted to divide these measurements into two clusters, and did so by fitting a mixture of two univariate Gaussians to the data.

The technique he used has been widely studied since, and is called *the method of moments*. Pick a function $t(\cdot)$, and compute its expectation under the empirical distribution of data points $x_1, \ldots, x_m$,

$$\widehat{\mathbf{E}}t = \frac{t(x_1) + \cdots + t(x_m)}{m}.$$

Now choose a model $\theta \in \Theta$ for which the expectation of t under θ is equal to $\widehat{\mathbf{E}}t$. Often a number of different functions t_i are chosen and the fit need not be exact.

Pearson chose the first few moment functions $t_i(x) = x^i$. He reduced the resulting equivalences to a single nonic equation whose solution would yield a mixture of Gaussians.

In scaling this technique to mixtures with more parameters, a few considerations are important. First, the functions $t(\cdot)$ should not have high variance, otherwise their empirical estimates are unreliable. This can sometimes rule out high-order moments, suggesting instead the “zero-order moments” $t_A(x) = 1(x \in A)$, for sets A. It seems difficult to find a set of moments to use with mixtures of Gaussians which will make the necessary equation-solving easy to automate. Perhaps this is why it appears to have died out as a viable algorithm, despite its intuitive appeal.

2.2 Expectation-maximization

The principal goal of the EM algorithm (Dempster, Laird and Rubin, 1977; Redner and Walker, 1984; Xu and Jordan, 1996) when applied to our learning problem is to find a local maximum in the space of Gaussian mixtures ranked by likelihood. We have already seen that the global optimum is not necessarily desirable, which puts the whole enterprise in question. Nonetheless EM has an appealing simplicity which, together with a marked absence of significantly better alternatives, has made it the current algorithm of choice for learning mixtures of Gaussians.

We will briefly sketch the algorithm. It starts by guessing some values for the mixture parameters, and then alternates between two stages.

> Stage ‘E’: Taking the current guess as truth, compute, for each data point and each Gaussian, the probability that the point came from that Gaussian. This gives a “soft clustering” of the data set – each point is not definitively assigned to a Gaussian, but is split between all of them in suitable proportion.
>
> Stage ‘M’: Based on this interim clustering, update the estimates of the parameters.

As this process continues, the log-likelihood of the estimated mixture increases monotonically. A possible stopping point is when the increase becomes very small, although of course this could just be a misleading lull.

One issue that has received a fair amount of attention is how the initial parameters should be chosen, particularly the initial Gaussian centers (for instance, Meilă and Heckerman, 1998). Viable options for the initial centers are: (i) choose a few of the data points at random; (ii) use the results of k-means, a quicker clustering heuristic; or (iii) find the bounding box of the data and choose some points uniformly at random from this box.

3 High-dimensional Gaussians

3.1 Some counter-intuitive effects

An n-dimensional Gaussian $N(\mu, \Sigma)$ has density function

$$p(x) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right).$$

If Σ is a multiple of the identity matrix, then the Gaussian is called *spherical*. Some important intuition about the behavior of Gaussians in high dimension can quickly be gained by examining the spherical Gaussian $N(0, \sigma^2 I_n)$. Although its density is highest at the origin, it turns out that for large n most of the probability mass lies far away from this center. This is the first of many surprises that high-dimensional space will spring upon us. A point $X \in \mathbb{R}^n$ chosen randomly from this Gaussian has coordinates X_i which are i.i.d. $N(0, \sigma^2)$. Therefore its expected squared Euclidean norm is $\mathbf{E}(\|X\|^2) = \sum_i \mathbf{E}X_i^2 = n\sigma^2$. In fact, it can be shown quite routinely, by writing out the moment-generating function of $\|X\|^2$, that the distribution of $\|X\|^2$ will be tightly concentrated around its expected value. Specifically,

$$\mathbf{P}(|\|X\|^2 - \sigma^2 n| > \epsilon\sigma^2 n) \quad \leq \quad 2e^{-n\epsilon^2/24}.$$

That is to say, for big enough n, almost the entire distribution lies in a thin shell of radius approximately $\sigma\sqrt{n}$. Thus the natural scale of this Gaussian is in units of $\sigma\sqrt{n}$.

This effect might arouse some initial skepticism because it is not observable in one or two dimensions. But it can perhaps be made more plausible by the following explanation. The Gaussian $N(0, I_n)$ assigns density proportional to $e^{-\rho^2 n/2}$ to points on the surface of the sphere centered at the origin and of radius $\rho\sqrt{n}, \rho \leq 1$. But the surface area of this sphere is proportional to $(\rho\sqrt{n})^{n-1}$. For large n, as $\rho \uparrow 1$, this surface area is growing much faster than the density is decaying, and thus most of the probability mass lies at distance about $\sqrt{n}$ from the origin. Figure 1 is a graphical depiction of this effect for various values of n.

The more general Gaussian $N(0, \Sigma)$ has ellipsoidal contours of equal density. Each such ellipsoid is of the form $\{x : x^T\Sigma^{-1}x = r^2\}$, corresponding to points at a fixed *Mahalanobis distance* $\|x\|_\Sigma = \sqrt{x^T\Sigma^{-1}x}$ from the center of the Gaussian. The principal axes of any of these ellipsoids are given by the eigenvectors of Σ. The radius along a particular axis is proportional to the square root of the corresponding eigenvalue. Denote the eigenvalues by

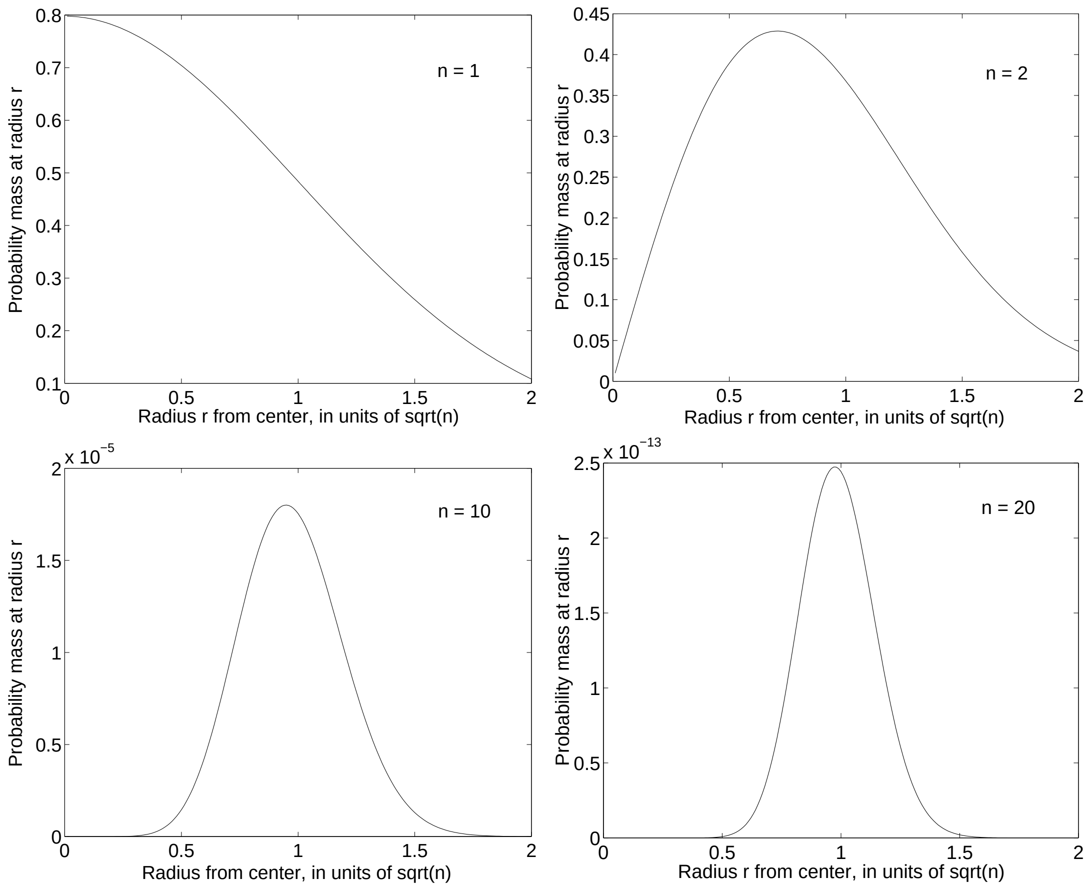


Figure 1: Probability mass of $N(0, I_n)$ at radius $r\sqrt{n}$ from the origin, for various n.

$\lambda_1 \leq \cdots \leq \lambda_n$. We will measure how non-spherical a Gaussian is by its *eccentricity*, namely $\sqrt{\lambda_n/\lambda_1}$. As in the spherical case, for large n and for Σ of bounded eccentricity the distribution of $N(0, \Sigma)$ will be concentrated around an ellipsoidal shell $\|x\|_\Sigma^2 \approx n$. Yet it will also be concentrated, perhaps less tightly, around a spherical shell $\|x\|^2 \approx \lambda_1 + \cdots + \lambda_n = \text{trace}(\Sigma)$.

3.2 Formalizing separation

It is reasonable to imagine, and is borne out by experience with techniques like EM (Duda and Hart, 1973; Redner and Walker, 1984), that a mixture of Gaussians is easiest to learn when the Gaussians do not overlap too much. Our discussion of $N(\mu, \sigma^2 I_n)$ suggests that it is natural to define the *radius* of this Gaussian as $\sigma\sqrt{n}$, which leads to the following

Definition Two Gaussians $N(\mu_1, \sigma^2 I_n)$ and $N(\mu_2, \sigma^2 I_n)$ are *c-separated* if $\|\mu_1 - \mu_2\| \geq c\sigma\sqrt{n}$. More generally, Gaussians $N(\mu_1, \Sigma_1)$ and $N(\mu_2, \Sigma_2)$ in $\mathbb{R}^n$ are c-separated if

$$\|\mu_1 - \mu_2\| \geq c\sqrt{n \max(\lambda_{max}(\Sigma_1), \lambda_{max}(\Sigma_2))},$$

where $\lambda_{max}(\Sigma)$ is shorthand for the largest eigenvalue of Σ. A mixture of Gaussians is c-separated if its component Gaussians are pairwise c-separated.

In other words, two spherical Gaussians are c-separated if their centers are c radii apart. In high dimension, a 2-separated mixture corresponds roughly to almost completely separated Gaussians, whereas a mixture that is 1- or $\frac{1}{2}$-separated has slightly more (though still negligible) overlap. The algorithm presented in the following chapter can deal with Gaussians which are arbitrarily close together; its running time, however, will inevitably depend on their level of separation.

One way to think about high-dimensional c-separated mixtures is to imagine that their projections to any one coordinate are c-separated. For instance, suppose that measurements are made on a population consisting of two kinds of fish. Various attributes, such as length and weight, are recorded. Suppose also that restricting attention to any one attribute gives a 1-separated mixture of two Gaussians in $\mathbb{R}^1$, which is unimodal and therefore potentially difficult to learn. However, if several (say ten) independent attributes are considered together, then the mixture in $\mathbb{R}^{10}$ will remain 1-separated but will no longer have a unimodal distribution. It is precisely to achieve such an effect that data sets generally try to incorporate as many relevant attributes as possible. This improvement in terms of better-defined clusters is bought at the price of an increase in dimensionality. Are there learning algorithms which can effectively exploit this tradeoff?

3.3 The problem of dimension

Why is it at all difficult to reconstruct a mixture of Gaussian given i.i.d. samples from it? In low dimension, for instance in the case of univariate Gaussians, it is often possible to simply plot the data and visually estimate a solution, provided the Gaussians maintain a respectable distance from one another. This is because a reasonable amount of data conveys a fairly accurate idea of the overall probability density. The high points of this density correspond to centers of Gaussians and to regions of overlap between neighboring clusters. If the Gaussians are far apart, these modes themselves provide good estimates of the centers.

Easy algorithms of this kind fail dismally in higher dimension. Consider again the Gaussian $N(\mu, \sigma^2 I_n)$. We must pick $2^{\Omega(n)}$ random points from this distribution in order to get just a few which are at distance $\leq \frac{1}{2}\sigma\sqrt{n}$ from the center. The data in any sample of plausible size, if plotted somehow, would resemble a few scattered specks of dust in an enormous void. What can we possibly glean from such a sample? Such gloomy reflections have prompted researchers to try mapping data into spaces of low dimension.

4 Dimensionality reduction

4.1 Overview

The naive mode-finding algorithm we just considered requires at least about 2^d data points to learn a mixture of Gaussians in $\mathbb{R}^d$. Is it possible to reduce the dimension of the data so dramatically that this requirement actually becomes reasonable?

Dimensionality reduction has been the subject of keen study for the past few decades, and instead of trying to summarize this work we will focus upon two very popular contemporary techniques: principal component analysis (PCA) and random projection. They are both designed for data with Euclidean (L_2) interpoint distances and are both achieved via linear mappings.

The linear projection of a Gaussian remains a Gaussian. Therefore, projecting a mixture of high-dimensional Gaussians onto a single line will produce a mixture of univariate Gaussians. However, these projected clusters might be so close together as to be indistinguishable. The main question then is, how much can the dimension be reduced while still maintaining a reasonable amount of separation between different clusters?

4.2 Principal component analysis

Principal component analysis is an extremely important tool for data analysis which has found use in many experimental and theoretical studies. It finds a d-dimensional subspace of $\mathbb{R}^n$ which captures as much of the variation in the data set as possible. Specifically, given data $S = \{x_1, \ldots, x_m\}$ it finds the linear projection to $\mathbb{R}^d$ for which

$$\sum_{i=1}^{m} \|x_i^* - \mu^*\|^2$$

is maximized, where x_i^* is the projection of point x_i and μ^* is the mean of the projected data.

This projection is quite easy to obtain. Let μ, Σ denote the mean and covariance of the high-dimensional data S. The positive semidefinite matrix Σ can be written in the form $B^T D B$, where $D = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ is a diagonal matrix containing the eigenvalues of Σ, and B is an orthogonal $n \times n$ matrix (that is, $B^T B = B B^T = I_n$). If the data points are rotated by B, the resulting data $Bx_1, \ldots, Bx_m$ has mean $B\mu$ and covariance

$$\frac{1}{m} \sum_{i=1}^{m} (Bx_i - B\mu)(Bx_i - B\mu)^T = B \Sigma B^T = D.$$

Assume the eigenvalues are ordered so that $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$. Then the direction of maximum variance of the rotated data is simply the first coordinate axis. Similarly, to select the d-dimensional subspace of maximum variance, simply pick the first d coordinate axes of the rotated space. In summary, PCA projects each point $x_i \in \mathbb{R}^n$ to $x_i^* = P^T x_i$, where P^T is the $d \times n$ projection matrix consisting of the first d rows of B. The projected data then has covariance $\mathrm{diag}(\lambda_1, \ldots, \lambda_d)$.

By how much can PCA reduce the dimension of a mixture of k Gaussians? It is quite easy to symmetrically arrange a group of k spherical Gaussians in $\mathbb{R}^{k/2}$ so that a PCA projection to any smaller dimension will collapse some of the Gaussians together, and thereby decisively derail any hope of learning. For instance, place the centers of the $(2j-1)^{st}$ and $2j^{th}$ Gaussians along the j^{th} coordinate axis, at positions j and $-j$. The eigenvectors found by PCA will roughly be coordinate axes, and the discarding of any eigenvector will collapse together the corresponding pair of Gaussians. Thus PCA cannot in general be expected to reduce the dimension of a mixture of k Gaussians to below $\Omega(k)$. Moreover, it is a rather time-consuming process for high-dimensional data.

4.3 Random projection

A much faster technique for dimensionality reduction, which has received a warm welcome in the theoretical computer science community, is expressed in the following lemma.

Lemma 1 (Johnson-Lindenstrauss, 1984) *Fix some* $0 < \epsilon < 1$ *and* $0 < \delta < 1$*. Pick any* m *data points* $x_1, \ldots, x_m \in \mathbb{R}^n$*. If these points are projected into a randomly chosen subspace of dimension* $d \geq \frac{12}{\epsilon^2} \ln \frac{m^2}{\delta}$ *then with probability* $> 1 - \delta$*, the projected points* $x_1^*, \ldots, x_m^*$ *satisfy*

$$(1-\epsilon)\,\frac{d}{n} \quad \leq \quad \frac{\|x_i^* - x_j^*\|^2}{\|x_i - x_j\|^2} \quad \leq \quad (1+\epsilon)\,\frac{d}{n} \qquad \textit{for all } i \neq j.$$

This remarkable lemma asserts, roughly, that any m data points in arbitrarily high dimension can be linearly mapped into an $O(\log m)$-dimensional subspace in such a way that pairwise Euclidean distances between the points are only slightly distorted. Moreover, this particular subspace is not hard to choose; a random subspace will with high probability ($> 1 - m^{-\Omega(1)}$) do the job. The original proof was simplified by Frankl and Maehara (1988), and a yet more elementary version, the result of joint work with Gupta, is presented in Appendix A.3.

The choice of random projection matrix does not depend upon the data in any way, and so can be accomplished quickly, in time $O(d^2 n)$. This technique is therefore a very efficient generic means of coping with high dimensionality. However, for our purposes this reduced dimension is still far too high. The rough heuristic outlined above needs 2^d data points, and this exceeds m by many orders of magnitude.

4.4 Random projection for mixtures of Gaussians

Later in this chapter we will see that *for the particular case of mixtures of Gaussians*, we can reduce the dimension of the data far more drastically than in the generic bound given above. By using projection to a randomly chosen subspace as in the Johnson-Lindenstrauss lemma, we can map the data into just $d = O(\log k)$ dimensions, where k is the number of Gaussians. Therefore the amount of data we will need is only polynomial in k.

This might puzzle readers who are familiar with random projection, because the usual motive behind such projections is to approximately preserve relative distances between data points. However, in our situation we expressly do not want this. We want some of the pairwise distances to contract significantly, so that the fraction of points within distance $\Delta\sqrt{d}$ of any Gaussian center in the reduced space $\mathbb{R}^d$ is much greater than the fraction of points within distance $\Delta\sqrt{n}$ of the same center in the original space $\mathbb{R}^n$. At the same time, we do not want the distances between different Gaussians to contract; we must make sure that Gaussians which are well-separated remain so when they are projected. These conflicting requirements are accommodated admirably by a projection to just $O(\log k)$ dimensions.

Definition For a positive definite matrix Σ, let $\lambda_{max}(\Sigma)$ and $\lambda_{min}(\Sigma)$ refer to its largest and smallest eigenvalues, respectively, and denote by $\mathrm{E}(\Sigma)$ the *eccentricity* of the matrix, that is, $\sqrt{\lambda_{max}(\Sigma)/\lambda_{min}(\Sigma)}$.

The following dimensionality reduction lemma applies to arbitrary mixtures of Gaussians, which we parametrize by mixing weights w_i, means μ_i and covariance matrices Σ_i, one per Gaussian. Its statement refers to the notion of separation introduced earlier.

Lemma 2 (Dimensionality reduction) *For any $c > 0$, let $\{(w_i, \mu_i, \Sigma_i)\}$ denote a c-separated mixture of k Gaussians in $\mathbb{R}^n$, and let $\delta, \epsilon \in (0,1)$ designate confidence and accuracy parameters, respectively. The projection of this mixture of Gaussians into a random d-dimensional subspace will with probability $> 1-\delta$ yield a $(c\sqrt{1-\epsilon})$-separated mixture of Gaussians $\{(w_i, \mu_i^*, \Sigma_i^*)\}$ in $\mathbb{R}^d$, provided $d \geq \frac{4}{\epsilon^2}\ln\frac{k^2}{2\delta}$.*

Moreover, $\lambda_{max}(\Sigma_i^*) \leq \lambda_{max}(\Sigma_i)$ and $\lambda_{min}(\Sigma_i^*) \geq \lambda_{min}(\Sigma_i)$. In particular therefore, $\mathrm{E}(\Sigma_i^*) \leq \mathrm{E}(\Sigma_i)$.

Proof. Consider a single line segment in $\mathbb{R}^n$, of squared length L. If the original space is projected onto a random d-dimensional subspace, the squared length of this line segment becomes some L^*, of expected value $\mathbf{E}L^* = Ld/n$. The proof of the Johnson-Lindenstrauss lemma (Appendix A.3) shows that $\mathbf{P}(L^* < (1-\epsilon)Ld/n) \leq e^{-d\epsilon^2/4}$.

Apply this bound to the $\binom{k}{2}$ line segments joining pairs of Gaussian centers in the original space. Then

$$\mathbf{P}\left(\exists i \neq j : \|\mu_i^* - \mu_j^*\| < \|\mu_i - \mu_j\|\sqrt{(1-\epsilon)d/n}\right) \quad \leq \quad \binom{k}{2}e^{-d\epsilon^2/4} \quad \leq \quad \delta.$$

This keeps the centers far apart; to satisfy our definition of separatedness, we must also check that the original Gaussians do not spread out when projected, that is, $\lambda_{max}(\Sigma_i^*) \leq \lambda_{max}(\Sigma_i)$.

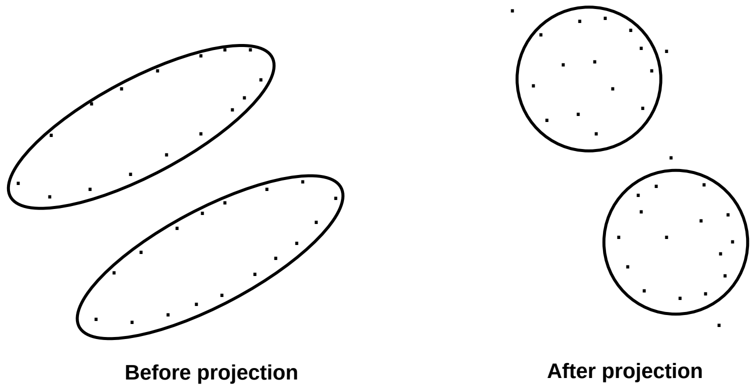


Figure 2: The effects of random projection: the dimension is drastically reduced while the clusters remain well-separated and become more spherical.

This is straightforward. Write the projection, say P^T, as a $d \times n$ matrix with orthogonal rows. P^T sends Gaussian $N(\mu, \Sigma)$ in $\mathbb{R}^n$ to $N(P^T\mu, P^T\Sigma P)$ in $\mathbb{R}^d$, and

$$\begin{aligned}\lambda_{max}(P^T\Sigma P) &= \max_{u\in\mathbb{R}^d} \frac{u^T(P^T\Sigma P)u}{u^Tu} = \max_{v\in\mathbb{R}^n} \frac{(P^Tv)^T(P^T\Sigma P)(P^Tv)}{(P^Tv)^T(P^Tv)} \\ &= \max_{v\in\mathbb{R}^n} \frac{(PP^Tv)^T\Sigma(PP^Tv)}{(PP^Tv)^T(PP^Tv)} \\ &\leq \max_{v\in\mathbb{R}^n} \frac{v^T\Sigma v}{v^Tv} = \lambda_{max}(\Sigma).\end{aligned}$$

The denominator in the second line uses $P^TP = I_d$. In similar fashion we can show that $\lambda_{min}(\Sigma_i^*) \geq \lambda_{min}(\Sigma_i)$, completing the proof. ▮

This method of projection has another tremendous benefit: we show that even if the original Gaussians are highly skewed (have ellipsoidal contours of high eccentricity), their projected counterparts will be more spherical (Figure 2). Since it is conceptually much easier to design algorithms for spherical clusters than ellipsoidal ones, this feature of random projection can be expected to simplify the learning of the projected mixture.

Consider a Gaussian $N(0, \Sigma)$ in $\mathbb{R}^n$, and think of the random projection from $\mathbb{R}^n$ to $\mathbb{R}^d$ as a random rotation in $\mathbb{R}^n$, represented by some orthogonal matrix U^T, followed by a projection P^T onto the first d coordinates. The columns of U^T are an orthonormal basis $\{u_1, \ldots, u_n\}$ of $\mathbb{R}^n$. Denote the restriction of these vectors to their first d coordinates by $u_1^*, \ldots, u_n^*$, respectively. The high-dimensional covariance matrix Σ has some eigenvalues $\lambda_1 \leq \cdots \leq \lambda_n$, with eccentricity $\textsc{e} = \sqrt{\lambda_n/\lambda_1} \geq 1$, and normalized trace $\lambda = \frac{1}{n}(\lambda_1 + \cdots + \lambda_n)$. It will turn out that under suitable conditions on $\textsc{e}$ the covariance matrix of the projected

Gaussians, denoted Σ^*, is close to the spherical covariance matrix λI_d.

Pick any unit vector $x \in \mathbb{R}^d$, and define $V(x) = x^T \Sigma^* x$ to be the variance of the projected Gaussian in direction x. The quantity $V(x)$ is a random variable which depends upon the choice of random projection. We next characterize its distribution.

Lemma 3 (Variance of a projected Gaussian) *For any unit vector $x \in \mathbb{R}^d$, $V(x)$ has the same distribution as $\sum_{i=1}^n \lambda_i v_i^2$, where v is chosen uniformly at random from the surface of the unit sphere in $\mathbb{R}^n$. Therefore $\mathbf{E}V(x) = \lambda$, over the choice of random projection.*

Proof. We can write the projected covariance matrix Σ^* as $(UP)^T\Sigma(UP)$, and on account of U we may assume Σ is diagonal, specifically $\Sigma = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$.

Pick any direction $x \in \mathbb{R}^d$. The variance of the projected Gaussian in direction x is $V(x) = x^T\Sigma^* x = (Px)^T(U^T\Sigma U)(Px)$. Since Σ is diagonal,

$$(U^T \Sigma U)_{ij} = \sum_{k=1}^n \lambda_k U_{ki} U_{kj}$$

whereby

$$\begin{aligned} V(x) &= \sum_{i,j=1}^n (Px)_i (Px)_j (U^T\Sigma U)_{ij} \\ &= \sum_{i,j=1}^d x_i x_j \sum_{k=1}^n \lambda_k U_{ki} U_{kj} \\ &= \sum_{k=1}^n \lambda_k \sum_{i,j=1}^d (x_i U_{ki})(x_j U_{kj}) \\ &= \sum_{k=1}^n \lambda_k (x \cdot u_k^*)^2, \end{aligned}$$

where u_k^* denotes the first d coordinates of the k^{th} row of U.

We can without loss of generality assume that x lies along some coordinate axis, say the very first one, in which case

$$V(x) = \sum_{i=1}^n \lambda_i u_{i1}^2.$$

Since U^T is a random orthogonal matrix, its first row $(u_{11}, \ldots, u_{n1})$ is a random unit vector. ∎

We now have a simple formulation of the distribution of $V(x)$. For any given x, this value is likely to be close to its expectation because it is the sum of n almost-independent bounded random variables. To demonstrate $V(x) \approx \lambda$ simultaneously for all vectors x on the unit sphere in $\mathbb{R}^d$, we will prove uniform convergence for a carefully chosen finite cover of this sphere.

Lemma 4 (Eccentricity reduction) *There is a universal constant C such that for any $0 < \epsilon \leq 1$ and $0 < \delta < 1$, if the original dimension satisfies $n > C \cdot \frac{\mathrm{E}^2}{\epsilon^2}(\log \frac{1}{\delta} + d \log \frac{d}{\epsilon})$, then with probability $> 1 - \delta$ over the choice of random projection, the eccentricity E^* of the projected covariance matrix will be at most $1 + \epsilon$. In particular, if the high-dimensional eccentricity E is at most $n^{1/2} C^{-1/2} (\log \frac{1}{\delta} + d \log d)^{-1/2}$ then with probability at least $1 - \delta$, the projected Gaussians will have eccentricity $\mathrm{E}^* \leq 2$.*

Proof. By considering moment-generating functions of various gamma distributions (see Lemma A.5 for details), we can show that for any particular x and any $\epsilon \in (0, 1]$,

$$\mathbf{P}(|V(x) - \lambda| > \tfrac{1}{6}\epsilon\lambda) \leq e^{-\Omega(n\epsilon^2/\mathrm{E}^2)}.$$

Moreover, if x and y are unit vectors which are close to each other then $V(y)$ cannot differ too much from $V(x)$:

$$\begin{aligned} |V(x) - V(y)| &\leq \sum_{i=1}^{n} \lambda_i \ \left|(u_i^* \cdot x)^2 - (u_i^* \cdot y)^2\right| \\ &= \sum_{i=1}^{n} \lambda_i \ |u_i^* \cdot (x - y)| \cdot |u_i^* \cdot (x + y)| \\ &\leq \sum_{i=1}^{n} \lambda_i \ \|u_i^*\|^2 \cdot \|x + y\| \cdot \|x - y\| \\ &\leq 2 \ \|x - y\| \left(\sum_{i=1}^{n} \lambda_i \|u_i^*\|^2 \right). \end{aligned}$$

The third line follows by the Cauchy-Schwarz inequality and the fourth uses the fact that x and y are unit vectors. The final parenthesized quantity has expected value $d\lambda$ (each u_i^* consists of the first d coordinates of a random unit vector in $\mathbb{R}^n$ and therefore $\mathbf{E}\|u_i^*\|^2 = \frac{d}{n}$). The chance that it exceeds $2d\lambda$ is at most $de^{-\Omega(n/\mathrm{E}^2)}$, by Lemma A.5. Choosing $\|x - y\| \leq \frac{\epsilon}{24d}$ will then ensure $|V(x) - V(y)| \leq \frac{1}{6}\epsilon\lambda$.

Bounding $V(x)$ effectively bounds $V(y)$ for $y \in B(x; \frac{\epsilon}{24d})$. How many points x must be chosen to cover the unit sphere in this way? It turns out that $(\frac{24d\beta}{\epsilon})^d$ points will do the trick, for some constant $\beta > 0$. This last result is shown by considering the dual question, how many non-overlapping spherical caps of fixed radius can be packed onto the surface of a sphere? An upper bound can be obtained by a surface area calculation (Gupta, 1999).

Let S denote this finite cover; then

$$\begin{aligned}\mathbf{P}(\mathrm{E}^* \geq 1+\epsilon) &\leq \mathbf{P}(\exists x : V(x) \not\in [\lambda(1-\tfrac{1}{3}\epsilon), \lambda(1+\tfrac{1}{3}\epsilon)]) \\ &\leq de^{-\Omega(n/\mathrm{E}^2)} + \mathbf{P}(\exists x \in S : V(x) \not\in [\lambda(1-\tfrac{1}{6}\epsilon), \lambda(1+\tfrac{1}{6}\epsilon)]) \\ &\leq de^{-\Omega(n/\mathrm{E}^2)} + (O(\tfrac{d}{\epsilon}))^d \cdot e^{-\Omega(n\epsilon^2/\mathrm{E}^2)},\end{aligned}$$

completing the proof. ▮

Random projection offers many clear benefits over principal component analysis. As explained earlier, PCA cannot in general be used to reduce the dimension of a mixture of k Gaussians to below $\Omega(k)$, whereas random projection can reduce the dimension to just $O(\log k)$. Moreover, PCA may not reduce the eccentricity of Gaussians. However, if a projection to k dimensions is acceptable, then a PCA-projected mixture could easily be far better separated than a randomly projected mixture. For this reason PCA remains an important tool in the study of Gaussian clusters.

4.5 Illustrative experiments

The lemmas of the previous section can be illustrated by a few simple experiments. The first two of these examine what happens when a 1-separated mixture of k Gaussians in $\mathbb{R}^n$ is randomly projected into $\mathbb{R}^d$. The main question is, in order to achieve a fixed level of separation in the projected mixture, what value of d must be chosen? How will this d vary with n and k?

The first experiment, depicted in Figure 3, is intended to demonstrate that d does not depend upon n, that is, the projected dimension is independent of the original dimension of the data. Here two 1-separated spherical Gaussians are projected into $\mathbb{R}^{20}$ and their separation is noted as a function of n. The error bars are for one standard deviation in either direction; there are 40 trials per value of n.

The second series of tests (Figure 4) randomly projects 1-separated mixtures of k spherical Gaussians in $\mathbb{R}^{100}$ into $d = 10 \ln k$ dimensions, and then notes the separation of the resulting mixtures. The results directly corroborate Lemma 2. The mixtures created for these tests are maximally packed, that is, each pair of constituent Gaussians is 1-separated. There are 40 measurements taken for each value of k.

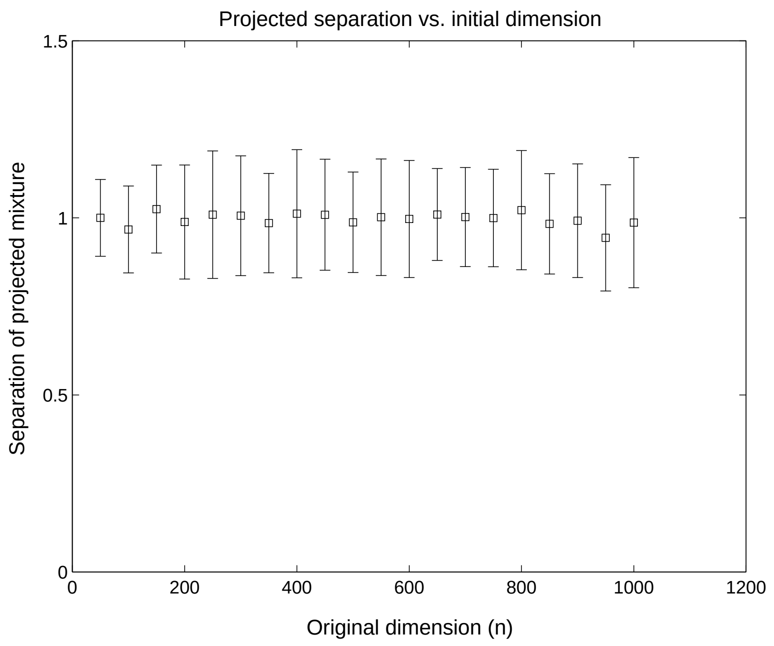


Figure 3: The projected dimension ($d = 20$) does not depend upon the original dimension.

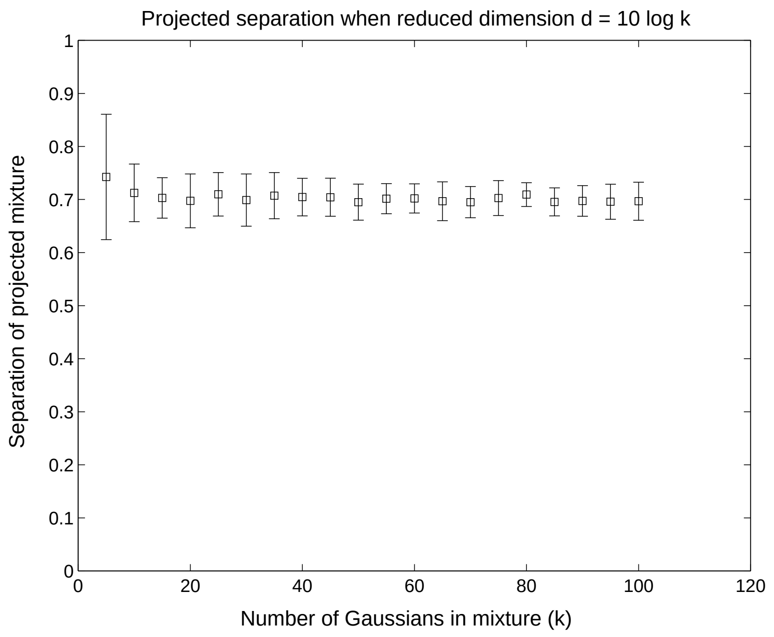


Figure 4: $d = 10 \ln k$ works nicely to keep the projected clusters well-separated.

Eccentricity E	n				
in $\mathbb{R}^n$	25	50	75	100	200
50	9.5 ± 3.80	3.4 ± 0.62	2.5 ± 0.29	2.2 ± 0.17	1.7 ± 0.07
100	13.1 ± 5.79	3.5 ± 0.57	2.5 ± 0.26	2.2 ± 0.19	1.7 ± 0.08
150	13.0 ± 7.40	3.5 ± 0.55	2.5 ± 0.25	2.2 ± 0.14	1.7 ± 0.07
200	14.7 ± 8.04	3.4 ± 0.50	2.5 ± 0.22	2.2 ± 0.19	1.7 ± 0.06

Figure 5: Reduction in eccentricity $\text{E} \rightarrow \text{E}^*$, for a variety of starting dimensions n. Values in the table represent eccentricities E^* in the projected space $\mathbb{R}^{20}$, plus or minus one standard deviation. Each table entry is the result of 40 trials.

The last two experiments, shown in Figures 5 and 6, document the dramatic decrease in eccentricity that accompanies the random projection of a Gaussian. The first of these projects a Gaussian of high eccentricity E from $\mathbb{R}^n$ into $\mathbb{R}^{20}$ and measures the eccentricity E^* of the projection, over a range of values of E and n. Here matrices of eccentricity E are constructed by sampling the square roots of their eigenvalues uniformly from the range $[1, \text{E}]$, and making sure to include the endpoints 1 and E. The last experiment fixes a particular Gaussian in $\mathbb{R}^{50}$ of enormous eccentricity $\text{E} = 1000$, and then projects this Gaussian into successively lower dimensions $49, 48, 47, \ldots, 25$. Notice that the y-axis of the graph has a logarithmic scale. Also, the error bars no longer represent standard deviations but in fact span the maximum and minimum eccentricities observed over 40 trials per value of d.

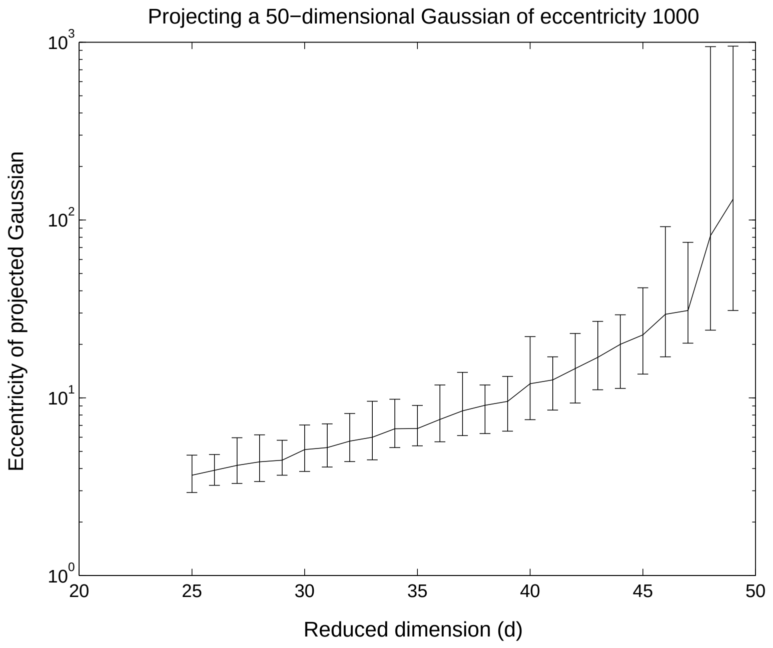


Figure 6: The eccentricity E^* in $\mathbb{R}^d$ of a Gaussian which has eccentricity $\mathrm{E} = 1000$ in $\mathbb{R}^{50}$.

Chapter III

A simple learning algorithm

1 Overview

In this chapter we will examine a simple algorithm which learns an unknown mixture of Gaussians with arbitrary common covariance matrix and arbitrary mixing weights, in time which scales only linearly with dimension and polynomially with the number of Gaussians. With high probability, it learns the true centers of the Gaussians to within the precision specified by the user. Previous heuristics have been unable to offer any such performance guarantee, even for highly restricted subcases like mixtures of two Gaussians.

The learning model was described in the introductory chapter (Figure I.6). The user furnishes: a data set S consisting of (say) m data points in $\mathbb{R}^n$; ϵ, the accuracy within which the centers are to be learned; δ, a confidence parameter; k, the number of Gaussians; and w_{min}, the smallest mixing weight that will be considered.

The algorithm, shown in Figure 1, works in four phases and is very simple to implement. Some brief intuition about its functioning will hopefully ease the way for the subsequent formal analysis.

1.1 Low-dimensional clustering

The data are projected from $\mathbb{R}^n$ to $\mathbb{R}^d$ via a linear map. Since any linear transformation of a Gaussian conveniently remains a Gaussian, we can pretend that the projected data themselves come from a mixture of low-dimensional Gaussians.

The second step of the algorithm estimates the means of these projected Gaussians.

1. **Projection** Select a random d-dimensional subspace of the original space $\mathbb{R}^n$, and project the data into this space. This step is motivated by the results of the previous chapter. Let S^* denote the projected data.

2. **Low-dimensional clustering** This step first estimates the low-dimensional centers and then finds data points near them. In the projected space,

   - For each $x \in S^*$, compute r_x, the smallest radius such that there are $\geq p$ points within distance r_x of x.
   - Start with $S' = S^*$.
   - For $i = 1 \ldots k$:
     - Let estimate $\widehat{\mu}_i^*$ be the point $x \in S'$ with the lowest r_x.
     - Find the q closest points to this estimated center.
     - Remove these points from S'.
   - For each i, let S_i^* denote the l points in S^* which are closest to $\widehat{\mu}_i^*$. Let S_i denote these same points in the original space.

3. **High-dimensional reconstruction** Let the (high-dimensional) estimate $\widehat{\mu}_i$ be the mean of S_i in $\mathbb{R}^n$.

4. **High-dimensional consolidation** Assign each data point in S to its closest $\widehat{\mu}_i$. Compute statistics of the resulting clusters to obtain estimates of the means, mixing weights, and covariance matrix.

Figure 1: An algorithm for learning mixtures of Gaussians. The parameters d, l, p, and q depend upon the inputs, and will be determined later.

Regions of higher density will tend to contain more points, and we can roughly imagine the density around any data point x to be inversely related to the radius r_x. In particular, the point with lowest r_x will be near the center of some (projected) Gaussian. If the Gaussians all share the same covariance, then this data point will most likely be close to the center of the Gaussian with highest mixing weight.

Once we have a good estimate for the center of one Gaussian, how do we handle the rest of them? The problem is that one Gaussian may be responsible for the bulk of the data if it has a particularly high mixing weight. All the data points with low r_x might come from this one over-represented Gaussian, and need to be eliminated from consideration somehow.

This is done by growing a wide region around the estimated center, and removing from contention all the points in it. The region should be large enough to remove all high-density points in that particular Gaussian, but should at the same time leave intact the high-density points of other Gaussians. The reader may wonder, how can we possibly know how large this region should be if we have no idea of either the covariance or the mixing weights? First, we pick the q points closest to the estimated center rather than using a preset radius; this accomplishes a natural scaling. Second, the probability of encountering a data point at a distance $\leq r$ from the center of the Gaussian grows exponentially with r, and this rapid growth tends to eclipse discrepancies of mixing weight and directional variance.

Both the techniques described – that of choosing the point with the next lowest r_x as a center-estimate, and then "subtracting" the points close to it – rely heavily on the accuracy of spherical density estimates. That is, they assume that for any sphere in $\mathbb{R}^d$, the number of data points which fall within that sphere is close to its expected value under the mixture distribution. That this is in fact the case follows from the happy circumstance that the concept class of spheres in $\mathbb{R}^d$ has VC dimension only $d + 1$.

1.2 Reconstruction in high-dimensional space

At this stage, projected centers in hand, we recall that our actual task was to find the Gaussian means in the original high-dimensional space. This is not too difficult conceptually. For each low-dimensional estimated center $\widehat{\mu}_i^*$, we pick the l data points closest to it in $\mathbb{R}^d$, call them S_i^*, and then average these same points in $\mathbb{R}^n$. We expect S_i^* to be relatively uncontaminated with points from other Gaussians (although we cannot of course avoid the odd straggler), and thus its mean in $\mathbb{R}^n$ should closely approximate μ_i.

The chief technical problem in the reconstruction is to show that small errors in the estimate $\widehat{\mu}_i^*$ are not grossly magnified when carried back into $\mathbb{R}^n$. The core question can be stated quite simply. Given that an unknown point $x \in \mathbb{R}^n$ drawn from Gaussian $N(0, \Sigma)$ gets projected to some $y \in \mathbb{R}^d$, what is the conditional distribution of $\|x\|$ given $\|y\|$? A bit of matrix analysis yields the answer.

1.3 Consolidating the clusters in high-dimensional space

The estimates obtained in the second and third phases of the algorithm will be very crude, of much lower precision than the user demands. Nevertheless, it can be shown that assigning each point in S to its closest high-dimensional center-estimate $\widehat{\mu}_i$ will cluster the data almost perfectly. Computing statistics of each cluster will then yield sharp estimates of its mean, covariance, and mixing weight.

How exactly did the projection help us? It enabled us to find, for each Gaussian, a set of data points drawn mostly from that Gaussian.

1.4 The main result

Theorem 1 *In what follows, C_1, C_2, and C_3 are universal constants. Suppose data is drawn from a mixture of k Gaussians in $\mathbb{R}^n$ which is 1-separated, has smallest mixing weight w_{min}, and has some common (unknown) covariance matrix Σ. If the user specifies confidence and accuracy parameters $\delta, \epsilon \in (0,1)$ respectively, then the algorithm will set the reduced dimension to some $d = C_1 \log \frac{1}{w_{min}\delta}$.*

Let σ_{max}^2 denote the maximum eigenvalue of Σ and let E denote its eccentricity. If (1) $\mathrm{E} \leq C_2 n^{1/2}(d \log d)^{-1/2}$, (2) $n \geq C_2(d + \log \frac{1}{\epsilon})$ and (3) the number of data points is $m \geq (\frac{1}{w_{min}\delta})^{C_3}$, then with probability $> 1 - \delta$, the center estimates returned by the algorithm will be accurate within L_2 distance $\epsilon \sigma_{max} \sqrt{n}$.

The algorithm runs in time $O(m^2 d + mdn)$, plus an additional $O(mn^2)$ if an estimate of the covariance matrix is sought.

This algorithm can in fact efficiently handle Gaussians whose separation is an arbitrarily small constant $c > 0$. It is only to curtail the proliferation of symbols that we insist upon 1-separation in this theorem. A similar comment applies to the eccentricity.

Mixtures of one-dimensional Gaussians cannot be handled by our procedure. Since it attempts to project data into $\Omega(\log k)$-dimensional space, the original dimension must be at least this high. The final phase of the algorithm obtains estimates of the means (and other parameters) using a *hard clustering* of the data, as in the k-means algorithm. Even a perfect hard clustering will result in an L_2 error of $O(e^{-n}\sqrt{n})$; this is tiny for large n but in our framework it necessitates the requirement $n \geq \Omega(\log \frac{1}{\epsilon})$. To remove this qualification a *soft clustering* can be used instead, as in the EM algorithm.

1.5 Notation

The following battery of notation will be used consistently through the chapter.

ϵ, δ	Accuracy and confidence, supplied by user
ϵ_0	Accuracy of spherical density estimates
m	Overall number of data points
n	Original dimension of data
d	Reduced dimension
k	Number of Gaussians
$w_i N(\mu_i, \Sigma)$	A mixture component (Gaussian) in $\mathbb{R}^n$
w_{min}	Lower bound on the w_i, supplied by user
c_{ij}	Separation between i^{th} and j^{th} Gaussians in $\mathbb{R}^n$
c	$\min_{i \neq j} c_{ij}$
c^*_{ij}, c^*	Separation of Gaussians in $\mathbb{R}^d$
$w_i N(\mu^*_i, \Sigma^*)$	Projection of i^{th} Gaussian into $\mathbb{R}^d$
$\pi^*(\cdot)$	Density of the entire projected mixture
$B(x; r)$	Sphere of radius r centered at x
$B(r'; r)$	$B(x; r)$ for some x with $\|x\| = r'$
l, p, q	Integer parameters needed by algorithm
ρ	The accuracy of phases two and three (a small constant)
S	Data set in $\mathbb{R}^n$
S^*	Data set in $\mathbb{R}^d$
$\sigma_{max}, \sigma_{min}$	$\sqrt{\lambda_{max}(\Sigma)}, \sqrt{\lambda_{min}(\Sigma)}$
E	Eccentricity $\sigma_{max}/\sigma_{min}$
$\sigma^*_{max}, \sigma^*_{min}, \mathrm{E}^*$	Similar, but in the projected space
$\nu(\cdot)$	$N(0, I_d)$
$\nu_{\Sigma^*}(\cdot)$	$N(0, \Sigma^*)$
T	A useful linear transformation in $\mathbb{R}^d$
$\|\cdot\|_\Sigma$	Mahalanobis distance, $\|x\|_\Sigma = \sqrt{x^T \Sigma^{-1} x}$
$E(z; r; \Sigma)$	Ellipsoid $\{x : \|x - z\|_\Sigma \leq r\}$

As we have already seen, we can think of E^* as a small constant even if E is large, and this will help us tremendously.

2 Projection

Theorem 2 *The first phase of the algorithm takes as input (1) a data set $S \subseteq \mathbb{R}^n$ and (2) k, the number of Gaussians (or an upper bound on it). It returns a projection of the data set, S^*. If c^*_{ij} denotes the separation between the i^{th} and j^{th} projected Gaussians, and E^* is the eccentricity of the projected covariance matrix,*

- *if $d \geq \frac{64}{9} \log \frac{k^2}{\delta}$ then with probability at least $1-\delta$, for all $i \neq j$, $c^*_{ij} \geq \frac{1}{2} c_{ij}$; and*
- *with probability at least $1-\delta$, for any $v > 0$,*
$$\mathrm{E}^* \leq 1 + \max\{v,\ C^{1/2} \cdot \frac{\mathrm{E}}{\sqrt{n}} \cdot \sqrt{\log \frac{1}{\delta} + d \log \frac{d}{v}}\}.$$
where C is the universal constant from Lemma II.4.

In particular, if the high-dimensional mixture is 1-separated, with Gaussians of eccentricity $\mathrm{E} < n^{1/2} C^{-1/2} (\log \frac{1}{\delta} + d \log d)^{-1/2}$, then with probability $> 1 - 2\delta$, its projection will be $\frac{1}{2}$-separated, with eccentricity $\mathrm{E}^* \leq 2$.

Proof. This follows immediately from the lemmas in Chapter II. ▮

3 Low-dimensional clustering

3.1 Technical overview

The second phase of the algorithm, which learns the centers of Gaussians in low dimension, is ad hoc though correct. The technical tools used in its analysis might be helpful in developing other similar routines, so we start with a brief overview of them.

The first question is, in what way is the Gaussian nature of the data being used? Using VC bounds, it can be shown that with just a modest number of samples, all spheres in $\mathbb{R}^d$ will contain roughly the expected number of points of under the mixture distribution. This is a convenient guarantee; we need no other control on the sampling error. More formally, the clustering algorithm will work under a

Weak Gaussian assumption For each sphere in $\mathbb{R}^d$, the fraction of S^* that falls in this sphere is the expected fraction under the mixture distribution, $\pm\epsilon_0$, where ϵ_0 reflects sampling error (and is in our case proportional to $m^{-1/2}$). The class of spheres may be substituted by some other class of small VC dimension.

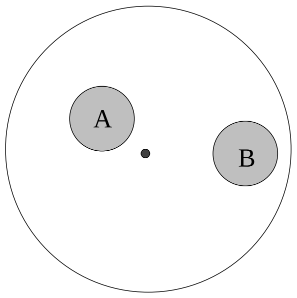


Figure 2: A sphere near the center of a Gaussian has higher probability mass than a sphere further away.

This requirement should be contrasted with a

Strong Gaussian assumption The data in S^* are drawn i.i.d. from the projected mixture of Gaussians.

The latter enables us to conclude, for instance, that if d is large and S^* consists of just two data points, then with high probability these two points are not close to one another. This does not follow from the weak assumption; in fact, the sampling error is so high for such a tiny data set that no non-trivial conclusions whatsoever can be drawn. The deliberate use of such a restriction is not mere conceit; it will turn out to be important for some of the extensions proposed later.

The weak Gaussian assumption forces us to examine the relative probability masses of different spherical regions in $\mathbb{R}^d$. Assume that the Gaussians are spherical. Each point x in the sample is assigned a radius r_x, and we hope that points with low r_x will be close to the centers of the Gaussians. In order to prove this, we must show that in cases such as that depicted in Figure 2 (where the outer sphere conceptually denotes a Gaussian), sphere A has a significantly higher probability mass than sphere B, which has the same radius but is further from the center of the Gaussian. This can be shown easily by a pointwise coupling of A and B.

Next, assume the Gaussians are spherical with unit variance. Once a center $\widehat{\mu}_i^*$ has been chosen, we will eliminate the q points closest to it, where q is the number of points expected to fall within $B(\mu_i^*; \frac{3}{8}\sqrt{d})$, assuming a mixing weight of w_{min}. It turns out that if $\widehat{\mu}_i^*$ is a reasonably accurate estimate of μ_i^* then whatever w_i might actually be, this process will eliminate all points in $B(\mu_i^*; \frac{1}{4}\sqrt{d})$ and nothing that does not lie in $B(\mu_i^*; \frac{1}{2}\sqrt{d})$. In effect, it eliminates all the high-density points in the i^{th} Gaussian while leaving intact

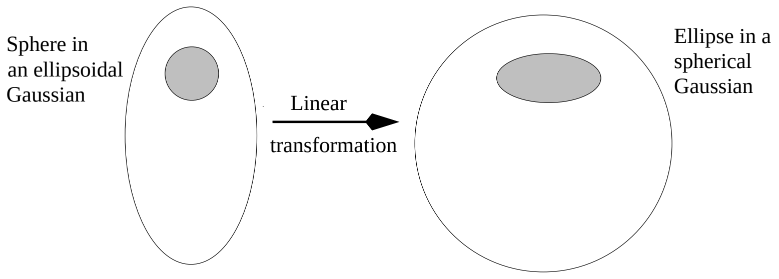

Sphere in an ellipsoidal Gaussian
Linear transformation
Ellipse in a spherical Gaussian

Figure 3: Estimating the probability mass of spherical regions in ellipsoidal Gaussians.

high-density regions of other Gaussians.

These arguments seem most naturally suited to spherical Gaussians. They all involve obtaining upper and lower bounds on the probability masses of spherical regions in $\mathbb{R}^d$. The extension to ellipsoidal Gaussians is managed via a simple linear transformation which maps a sphere contained in an ellipsoidal Gaussian to an ellipsoid contained in a spherical Gaussian. Bounds on the probability mass of this latter ellipsoid are then obtained by considering its inscribed and circumscribed spheres (Figure 3). These bounds are acceptable because the projected Gaussians have small eccentricity.

Once we have obtained estimates of the centers in $\mathbb{R}^d$, we pick a few points S_i^* close to each center-estimate $\widehat{\mu}_i^*$. We will need to argue that the points S_i^* were for the most part originally drawn from the i^{th} Gaussian G_i. In other words, we are thinking of data points as having hidden labels revealing which Gaussian generated them. A labelled point can be generated by the process

- Pick Gaussian G_i with probability w_i.
- Pick a point x from G_i, and label it with i.

or by the equivalent

- Pick Gaussian G_j with probability w_j.
- Pick a point x from G_j.
- Pick label i with probability proportional to ($w_i \cdot$ density assigned to x by G_i).

By switching between these two options, we will avoid various unpleasant dependencies between the data set and the estimated centers.

3.2 Summary

Theorem 3 *The second phase of the algorithm takes as input a (projected) data set $S^* \subseteq \mathbb{R}^d$ which satisfies the weak Gaussian assumption with error ϵ_0 with respect to a $\frac{1}{2}$-separated mixture of Gaussians $\{(w_i, \mu_i^*, \Sigma^*)\}$. The other inputs are: k, the number of Gaussians; the eccentricity E^*, or some upper bound on it; an accuracy parameter $\rho > 0$; and an integer l. The output of this phase is a set of k subsets $\{S_1^*, \ldots, S_k^*\} \subseteq S^*$, each of size l.*

The quantity ρ should be thought of as a small constant which is typically unrelated to the final precision ϵ required by the user. If the following conditions are met,

- $\rho \leq \min\{\frac{1}{16\mathrm{E}^*}, \frac{1}{8\mathrm{E}^{*2}}\}$,
- $\epsilon_0 \leq w_{min}(\frac{\rho}{8\mathrm{E}^*})^d \min\{\frac{1}{8}, \frac{3}{1024}\rho^2 d\}$,
- $d \geq \max\left\{38\mathrm{E}^{*4} \ln \frac{1}{w_{min}\rho^2}, \frac{8}{e} + 16 \ln \frac{10}{\rho w_{min}}\right\}$,
- $l \leq m w_{min}(\frac{1}{2}\rho)^d - m\epsilon_0$, *and*
- $l \geq \frac{30}{w_{min}\rho} \ln \frac{k^2}{\delta}$,

then with probability at least $1 - \delta$, each S_i^* is contained within $B(\mu_i^*; \rho\sigma_{max}^*\sqrt{d})$ and the number of points in S_i from the j^{th} Gaussian $G_j, j \neq i$, is $\leq \frac{lw_j\rho}{10c_{ij}^*}$.

Proof. This is a consequence of Lemmas 4 to 13, which follow. ∎

3.3 Crude density estimates

Our algorithm relies heavily upon the hope that in the projected space, every spherical region will contain roughly its expected number of points under the mixture distribution. This can shown effortlessly by a VC dimension argument.

Lemma 4 (Accuracy of density estimates) *Let $\nu(\cdot)$ denote any density on $\mathbb{R}^d$ from which i.i.d. data is drawn. If the number of data points seen satisfies $m \geq O\left(\frac{d}{\epsilon_0^2} \ln \frac{1}{\delta\epsilon_0}\right)$, then with probability $> 1 - \delta$, for every sphere $B \subset \mathbb{R}^d$, the empirical probability of that sphere differs from $\nu(B)$ by at most ϵ_0; that is, the number of points that fall in B is in the range $m\nu(B) \pm m\epsilon_0$.*

Proof. For any closed ball $B \subset \mathbb{R}^d$, let $1_B(x) = \mathbf{1}(x \in B)$ denote the indicator function for B. The concept class $\{1_B : B \subset \mathbb{R}^d \text{ is a sphere}\}$ has VC dimension $d+1$ (Dudley, 1979). The rest follows from well-known results about sample complexity; details can be found, for instance, in an article by Haussler (1992) or in the book by Pach and Agarwal (1995). ▮

We will henceforth assume that m meets the conditions of this lemma and that all spherical density estimates are accurate within ϵ_0. The next problem we face is that because Gaussians in general have ellipsoidal contours, it is not easy to get tight bounds on the probability mass of a given spherical region. We will content ourselves with rather loose bounds, obtained via the mediation of a linear transformation T which converts ellipsoids into spheres.

Write the $d \times d$ covariance matrix Σ^* as $B^T D B$, where B is orthogonal and D is diagonal with the eigenvalues of Σ^* as entries. Define $T = B^T D^{-1/2} B$; notice that T is its own transpose. The table below hints at the uses to which T shall be put.

In $\mathbb{R}^d$ before T is applied	In $\mathbb{R}^d$ after T is applied
Gaussian $N(\mu^*, \Sigma^*)$	Gaussian $N(T\mu^*, I_d)$
Point x, with $\|x\|_{\Sigma^*} = r$	Point Tx, with $\|Tx\| = r$
Ellipsoid $E(z; r; \Sigma^*)$	Sphere $B(Tz; r)$

Our first step will be to relate the density ν_{Σ^*} of the ellipsoidal Gaussian $N(0, \Sigma^*)$ to the more manageable density ν of $N(0, I_d)$.

Lemma 5 (Approximate density estimates for ellipsoidal Gaussians) *Choose any point z and any radius r. Writing $s = \|z\|_{\Sigma^*}$, the probability mass $\nu_{\Sigma^*}(B(z; r))$ must lie in the range $[\nu(B(s; \frac{r}{\sigma^*_{max}})), \nu(B(s; \frac{r}{\sigma^*_{min}}))]$.*

Proof. This is easy if T is used appropriately. For instance, since $E(z; \frac{r}{\sigma^*_{max}}; \Sigma^*) \subseteq B(z; r)$ we can write

$$\nu_{\Sigma^*}(B(z; r)) \quad \geq \quad \nu_{\Sigma^*}(E(z; \tfrac{r}{\sigma^*_{max}}; \Sigma^*)) \quad = \quad \nu(B(s; \tfrac{r}{\sigma^*_{max}})),$$

where the final equality is a result of applying the transformation T. ▮

Similarly we can bound the relative densities of displaced spheres. Consider two spheres of equal radius r, one close to the center of the Gaussian, at Mahalanobis distance s, and the other at some distance $s + \Delta$. By how much must the probability mass of the closer sphere exceed that of the farther one, given that they may lie in different directions from the center? Although the spheres have equal radius, it might be the case that the closer sphere lies in a direction of higher variance than the farther sphere, in which case its radius is effectively scaled down. The following lemma gives a bound that will work for all spatial configurations of the spheres.

Lemma 6 *Pick any point z and set $s = \|z\|_{\Sigma^*}$. If $\|z'\|_{\Sigma^*} \geq s + \Delta$ for some $\Delta > 0$ and if radius $r \leq s\sigma^*_{max}$ then*

$$\frac{\nu_{\Sigma^*}(B(z;r))}{\nu_{\Sigma^*}(B(z';r))} \geq \exp\left\{\frac{(\Delta+2s)(\Delta-2s\mathrm{E}^*)}{2}\right\}.$$

Proof. We will use the fact that Mahalanobis distance satisfies the triangle inequality and that $\|u\|_{\Sigma^*} \leq \|u\|/\sigma^*_{min}$. For any point x in $B(z;r)$,

$$\|x\|_{\Sigma^*} \;\leq\; \|z\|_{\Sigma^*} + \|x-z\|_{\Sigma^*} \;\leq\; s + \tfrac{r}{\sigma^*_{min}} \;\leq\; s + s\mathrm{E}^*,$$

where the last inequality follows from our restriction on r. Similarly, for any point x' in $B(z';r)$,

$$\|x'\|_{\Sigma^*} \;\geq\; \|z'\|_{\Sigma^*} - \|x'-z'\|_{\Sigma^*} \;\geq\; \Delta - s(\mathrm{E}^*-1).$$

Since $\nu_{\Sigma^*}(y)$ is proportional to $\exp(-\|y\|^2_{\Sigma^*}/2)$ for any point y, the ratio of probabilities of the two spheres must be at least

$$\frac{e^{-(s(1+\mathrm{E}^*))^2/2}}{e^{-(\Delta-s(\mathrm{E}^*-1))^2/2}} = \exp\left\{\frac{(\Delta-2s\mathrm{E}^*)(\Delta+2s)}{2}\right\},$$

as anticipated. ▌

Finally we need a bound on the rate at which the probability mass of a sphere, under distribution ν_{Σ^*}, grows as its radius increases.

Lemma 7 *If radii r and s satisfy $r + s \leq \frac{1}{2}\sigma^*_{min}\sqrt{d}$ then*

$$\frac{\nu_{\Sigma^*}(B(0;r+s))}{\nu_{\Sigma^*}(B(0;r))} \geq \left(\frac{r+s}{r}\right)^{d/2}.$$

Proof. Notice that

$$\nu_{\Sigma^*}(B(0;r)) \;=\; \int_{B(0;r)} \nu_{\Sigma^*}(x)dx \;=\; \left(\frac{r}{r+s}\right)^d \int_{B(0;r+s)} \nu_{\Sigma^*}\left(y \cdot \frac{r}{r+s}\right) dy$$

via the change in variable $y = x \cdot \frac{r+s}{r}$. Therefore

$$\frac{\nu_{\Sigma^*}(B(0;r+s))}{\nu_{\Sigma^*}(B(0;r))} = \left(\frac{r+s}{r}\right)^d \frac{\int_{B(0;r+s)} \nu_{\Sigma^*}(y)dy}{\int_{B(0;r+s)} \nu_{\Sigma^*}(y \cdot \frac{r}{r+s})dy}.$$

We will bound this ratio of integrals by considering a pointwise ratio. For any $y \in B(0; r+s)$, we know $\|y\|_{\Sigma^*} \leq (r+s)/\sigma^*_{min}$ and so

$$\begin{aligned} \frac{\nu_{\Sigma^*}(y)}{\nu_{\Sigma^*}(y \cdot \frac{r}{r+s})} &= \exp\left\{ -\frac{\|y\|^2_{\Sigma^*}}{2}\left(1 - \frac{r^2}{(r+s)^2}\right)\right\} \\ &\geq \exp\left\{ -\frac{(r+s)^2 - r^2}{2\sigma^{*2}_{min}}\right\} \\ &\geq \left(\frac{r}{r+s}\right)^{d/2}, \end{aligned}$$

given the condition on $r+s$. ▮

We next examine a few technical properties of the density ν of $N(0, I_d)$, as a step towards showing that there are many data points near the centers of projected Gaussians.

Lemma 8 (Crude lower bounds) *If $\tau \leq \frac{1}{3}$ and $d \geq 10$,*
(a) $\nu(B(0; \tau\sqrt{d})) \geq \tau^d$, and (b) $\nu(B(\tau\sqrt{d}; \tau\sqrt{d})) \geq \tau^d$.

Proof. Let V_d denote the volume of the unit ball in d dimensions. We will use the lower bound $V_d = \frac{\pi^{d/2}}{\Gamma(1+d/2)} \geq \frac{(2\pi)^{d/2}}{2(d/2)^{d/2}}$ which follows from the observation $\Gamma(1+k) \leq k^k 2^{-(k-1)}$ for $k \geq 1$. Now center a sphere at the mean of the Gaussian. A crude bound on its probability mass is

$$\nu(B(0; \tau\sqrt{d})) \quad \geq \quad \frac{e^{-(\tau\sqrt{d})^2/2}}{(2\pi)^{d/2}} \cdot V_d(\tau\sqrt{d})^d \quad \geq \quad \tau^d.$$

Continuing in the same vein, this time for a displaced sphere, we get bound (b). ▮

3.4 Estimating the projected centers

We are now in a position to prove that for an appropriate choice of the parameters p and q, the algorithm will find one data point close to each projected center, within Mahalanobis distance $\frac{1}{2}\rho\sqrt{d}$. This value ρ should be thought of as a small constant which is unrelated to (and typically much larger than) the precision ϵ demanded by the user. Denote by μ^*_i the means of the projected Gaussians and by Σ^* their common covariance matrix. Let π^* be the density of the projected mixture.

Parameter settings The parameters of the algorithm will be set to the following easily computable values,

- $p = m(w_{min}(\frac{\rho}{8\mathrm{E}^*})^d - \epsilon_0)$, and

- $q = mw_{min}\nu(B(0; \frac{3}{8\mathrm{E}^*}\sqrt{d}))$.

Lemma 9 *There is at least one data point within Mahalanobis distance $\frac{\rho}{8\mathrm{E}^*}\sqrt{d}$ of each center. Any such point x has at least p data points close by, in $B(x; (\frac{\rho}{8\mathrm{E}^*})\sigma^*_{max}\sqrt{d})$, and thus $r_x \leq (\frac{\rho}{8\mathrm{E}^*})\sigma^*_{max}\sqrt{d}$.*

Proof. Since all the density estimates are accurate within ϵ_0, we need only show that $w_{min}\nu_{\Sigma^*}(E(0; (\frac{\rho}{8\mathrm{E}^*})\sqrt{d}; \Sigma^*)) \geq \epsilon_0$ and that $w_{min}\nu_{\Sigma^*}(B(x; (\frac{\rho}{8\mathrm{E}^*})\sigma^*_{max}\sqrt{d})) \geq \frac{p}{m} + \epsilon_0$ for points x with $\|x\|_{\Sigma^*} \leq (\frac{\rho}{8\mathrm{E}^*})\sqrt{d}$. Transformation T and Lemma 5 convert statements about ν_{Σ^*} into statements about ν, in particular,

$\nu_{\Sigma^*}(E(0; (\frac{\rho}{8\mathrm{E}^*})\sqrt{d}; \Sigma^*)) = \nu(B(0; (\frac{\rho}{8\mathrm{E}^*})\sqrt{d}))$ and

$\nu_{\Sigma^*}(B(x; (\frac{\rho}{8\mathrm{E}^*})\sigma^*_{max}\sqrt{d})) \geq \nu(B((\frac{\rho}{8\mathrm{E}^*})\sqrt{d}; (\frac{\rho}{8\mathrm{E}^*})\sqrt{d}))$.

By Lemma 8 it is enough to check that $w_{min}(\frac{\rho}{8\mathrm{E}^*})^d \geq \frac{p}{m} + \epsilon_0$. This follows from the choice of p. ▮

This lemma gives an upper bound on r_x for points x close to a center. We next need to show that r_x will be significantly larger for points further away, at Mahalanobis distance $\geq \frac{1}{2}\rho\sqrt{d}$ from the center.

Lemma 10 *Suppose $r_x \leq (\frac{\rho}{8\mathrm{E}^*})\sigma^*_{max}\sqrt{d}$ for some point x which is at Mahalanobis distance $\geq \frac{1}{2}\rho\sqrt{d}$ from the closest center μ^*_i and at L_2 distance $\geq \frac{1}{4\mathrm{E}^*}\sigma^*_{min}\sqrt{d}$ from all other centers.*

Then, if $d \geq 38\mathrm{E}^{*4}\ln\frac{1}{w_{min}\rho^2}$ and $\rho \leq \min\{\frac{1}{16\mathrm{E}^*}, \frac{1}{8\mathrm{E}^{*2}}\}$, any point z within Mahalanobis distance $\frac{\rho}{8\mathrm{E}^*}\sqrt{d}$ of μ^*_i will have $r_z < r_x$.

Proof. The conditions on x imply that

(1) $\|x - \mu^*_i\|_{\Sigma^*} \geq \frac{1}{2}\rho\sqrt{d}$;

(2) $\|x - \mu^*_j\|_{\Sigma^*} \geq \frac{1}{4\mathrm{E}^*}\frac{\sigma^*_{min}}{\sigma^*_{max}}\sqrt{d} = \frac{1}{4\mathrm{E}^{*2}}\sqrt{d}$ for $j \neq i$; and

(3) $\pi^*(B(x; r_x)) \geq \frac{p}{m} - \epsilon_0$.

Now pick any z within Mahalanobis distance $\frac{\rho}{8\mathrm{E}^*}\sqrt{d}$ of μ^*_i. We need to show that $\pi^*(B(z; r_x)) \geq 2\epsilon_0 + \pi^*(B(x; r_x))$. By Lemma 6,

$$\begin{aligned}
\pi^*(B(x; r_x)) &= w_i\nu_{\Sigma^*}(B(x - \mu^*_i; r_x)) + \sum_{j\neq i} w_j\nu_{\Sigma^*}(B(x - \mu^*_j; r_x)) \\
&\leq w_i\nu_{\Sigma^*}(B(z - \mu^*_i; r_x))\exp(-\tfrac{1}{2}(\tfrac{1}{2}\rho + \tfrac{\rho}{8\mathrm{E}^*})(\tfrac{1}{4}\rho - \tfrac{\rho}{8\mathrm{E}^*})d) \\
&\quad + \nu_{\Sigma^*}(B(z - \mu^*_i; r_x))\exp(-\tfrac{1}{2}(\tfrac{1}{4\mathrm{E}^{*2}} + \tfrac{\rho}{8\mathrm{E}^*})(\tfrac{1}{4\mathrm{E}^{*2}} - \tfrac{\rho}{8\mathrm{E}^*} - \tfrac{1}{4}\rho)d) \\
&\leq w_i\nu_{\Sigma^*}(B(z - \mu^*_i; r_x))F \\
&\leq \pi^*(B(z; r_x))F
\end{aligned}$$

where $F \stackrel{\text{def}}{=} \exp(-\frac{d}{2}(\frac{1}{2}\rho + \frac{\rho}{8\mathrm{E}^*})(\frac{1}{4}\rho - \frac{\rho}{8\mathrm{E}^*})) + \frac{1}{w_{min}}\exp(-\frac{d}{2}(\frac{1}{4\mathrm{E}^{*2}} + \frac{\rho}{8\mathrm{E}^*})(\frac{1}{4\mathrm{E}^{*2}} - \frac{\rho}{8\mathrm{E}^*} - \frac{1}{4}\rho))$. Therefore, using (3),

$$\pi^*(B(z;r_x)) - \pi^*(B(x;r_x)) \;\geq\; \pi^*(B(x;r_x))\frac{1-F}{F} \;\geq\; \left(\frac{p}{m} - \epsilon_0\right)\frac{1-F}{F}.$$

Due to the particular choice of p and ϵ_0, this will be $\geq 2\epsilon_0$ if $1 - F \geq \min(\frac{1}{4}, \frac{3}{512}\rho^2 d)$. The latter is ensured by the lower bound on d. ∎

Lemma 10 implies roughly that within any Gaussian, the lowest r_x values come from data points which are within distance $\frac{1}{2}\rho\sqrt{d}$ of the center.

A potential problem is that a few of the Gaussians might have much higher mixing weights than the rest and consequently have a monopoly over small r_x values. In order to handle this, after selecting a center-estimate we eliminate the q points closest to it, and guarantee that this knocks out the high-density points near the current center while leaving intact the high-density regions near other centers.

Lemma 11 *Assume $\rho \leq \min\{\frac{1}{16\mathrm{E}^*}, \frac{1}{8\mathrm{E}^{*2}}\}$ and $d \geq 11 \ln \frac{2}{w_{min}}$. Let x be any point within Mahalanobis distance $\frac{1}{2}\rho\sqrt{d}$ of some center μ_i^*. Then the q data points closest to x include all data points in $B(\mu_i^*; \frac{1}{4\mathrm{E}^*}\sigma^*_{min}\sqrt{d})$ and no point outside $B(\mu_i^*; (\frac{1}{2\mathrm{E}^*} - \frac{\rho}{8\mathrm{E}^*})\sigma^*_{max}\sqrt{d})$.*

Proof. Rewriting $\frac{q}{m}$ as $w_{min}\nu_{\Sigma^*}(E(0; \frac{3}{8\mathrm{E}^*}\sqrt{d}; \Sigma^*))$, we notice that

$$w_{min}\nu_{\Sigma^*}(B(0; \tfrac{3}{8\mathrm{E}^*}\sigma^*_{min}\sqrt{d})) \leq \frac{q}{m} \leq w_{min}\nu_{\Sigma^*}(B(0; \tfrac{3}{8\mathrm{E}^*}\sigma^*_{max}\sqrt{d})).$$

For the first inclusion of the lemma, it is enough to show that $B(x; (\frac{1}{4\mathrm{E}^*} + \frac{\rho\mathrm{E}^*}{2})\sigma^*_{min}\sqrt{d})$, which contains $B(\mu_i^*; \frac{1}{4\mathrm{E}^*}\sigma^*_{min}\sqrt{d})$, has fewer than q points in it, or more specifically that

$$\pi^*(B(x; (\tfrac{1}{4\mathrm{E}^*} + \tfrac{\rho\mathrm{E}^*}{2})\sigma^*_{min}\sqrt{d})) \leq \frac{q}{m} - \epsilon_0$$

The left-hand side of this expression is maximized by choosing $x = \mu_i$ and $w_i = 1$. Moreover the choice of ϵ_0, q forces $\epsilon_0 \leq \frac{q}{2m}$; therefore it is enough to check that

$$\nu_{\Sigma^*}(B(0; (\tfrac{1}{4\mathrm{E}^*} + \tfrac{\rho\mathrm{E}^*}{2})\sigma^*_{min}\sqrt{d})) \leq \tfrac{1}{2}w_{min}\nu_{\Sigma^*}(B(0; \tfrac{3}{8\mathrm{E}^*}\sigma^*_{min}\sqrt{d})),$$

which is a direct consequence of Lemma 7 if $\rho \leq \frac{1}{8\mathrm{E}^{*2}}$ and $d \geq 11 \ln \frac{2}{w_{min}}$. The second half of the proof is executed similarly, and requires $\rho \leq \frac{1}{16\mathrm{E}^*}$ and $d \geq 10$. ∎

Lemma 12 (Accuracy of low-dimensional center estimates) *Under the assumptions stated in Theorem 3, the learned centers $\{\widehat{\mu}_i^*\}$ can be permuted so that for every $i \leq k$, $\|\widehat{\mu}_i^* - \mu_i^*\|_{\Sigma^*} \leq \frac{1}{2}\rho\sqrt{d}$.*

Proof. This will proceed by induction on the number of centers selected so far.

Referring back to the algorithm, the first center-estimate chosen is the point $x \in S$ with lowest r_x. By Lemma 9, this $r_x \leq (\frac{\rho}{8\mathrm{E}^*})\sigma^*_{max}\sqrt{d}$. Let μ^*_i be the projected center closest to x. Since the Gaussians are $\frac{1}{2}$-separated, x is at distance at least $\frac{1}{4}\sigma^*_{min}\sqrt{d}$ from all the other projected centers. By Lemma 10, we then see that x must be within Mahalanobis distance $\frac{1}{2}\rho\sqrt{d}$ of μ^*_i.

Say that at some stage in the algorithm, center-estimates $\widehat{C}$ have already been chosen, $|\widehat{C}| \geq 1$, and that these correspond to true centers C. Select any $y \in \widehat{C}$; by the induction hypothesis there is a j for which $\|y - \mu^*_j\|_{\Sigma^*} \leq \frac{1}{2}\rho\sqrt{d}$. S' does *not* contain the q points closest to y. By Lemma 11, this removes $B(\mu^*_j; \frac{1}{4\mathrm{E}^*}\sigma^*_{min}\sqrt{d})$ from S', yet no point outside $B(\mu^*_j; (\frac{1}{2\mathrm{E}^*} - \frac{\rho}{8\mathrm{E}^*})\sigma^*_{max}\sqrt{d})$ is eliminated from S' on account of y.

Let z be the next point chosen, and let μ^*_i be the center closest to it which is not in C. We have seen that z must be at distance at least $\frac{1}{4\mathrm{E}^*}\sigma^*_{min}\sqrt{d}$ from centers in C. Because of the separation of the mixture, z must be at distance at least $\frac{1}{4}\sigma^*_{min}\sqrt{d}$ from all other centers except possibly μ^*_i. Again due to the separation of the Gaussians, all points within distance $(\frac{\rho}{8\mathrm{E}^*})\sigma^*_{max}\sqrt{d}$ of μ^*_i remain in S', and therefore z is potentially one of these, whereupon, by Lemma 9, $r_z \leq (\frac{\rho}{8\mathrm{E}^*})\sigma^*_{max}\sqrt{d}$. By Lemma 10 then, $\|z - \mu^*_i\|_{\Sigma^*} \leq \frac{1}{2}\rho\sqrt{d}$. ▮

3.5 Choosing sets around the center-estimates

Once a center-estimate $\widehat{\mu}^*_i$ has been obtained, the l points $S^*_i \subset S^*$ closest to it are selected. Since by assumption $\frac{l}{m} \leq w_{min}(\frac{1}{2}\rho)^d - \epsilon_0$, it follows from Lemmas 5 and 8 that $S^*_i \subset B(\widehat{\mu}^*_i; \frac{1}{2}\rho\sigma^*_{max}\sqrt{d}) \subset B(\mu^*_i; \rho\sigma^*_{max}\sqrt{d})$. We must now show that S^*_i contains very little contamination from other Gaussians.

After the center-estimates have been chosen, label each data point by the true Gaussian from which it was drawn (that is, choose a label based on the probability of the Gaussian given the data point). This labelling is of course independent of the center-estimates. Let l_{ij} be the number of points in S^*_i which are labelled j. We need to show that l_{ij} is very small when $i \neq j$.

Lemma 13 *Under the assumptions of Theorem 3, with probability at least $1 - \delta$, for all $i \neq j$,*

$$\frac{l_{ij}}{l} \leq \frac{w_j\rho}{10c^*_{ij}}.$$

Proof. Pick any $x \in S_i^*$ and any $j \neq i$. We know $\|x - \mu_i^*\| \leq \rho\sigma_{max}^*\sqrt{d}$ and therefore $\|x - \mu_j^*\| \geq c_{ij}^*\sigma_{max}^*\sqrt{d} - \rho\sigma_{max}^*\sqrt{d}$. Then, applying the general principle that $\frac{\|u\|}{\sigma_{max}^*} \leq \|u\|_{\Sigma^*} \leq \frac{\|u\|}{\sigma_{min}^*}$, we can conclude

$$\|x - \mu_i^*\|_{\Sigma^*} \leq \mathrm{E}^*\rho\sqrt{d} \;\text{ and }\; \|x - \mu_j^*\|_{\Sigma^*} \geq (c_{ij}^* - \rho)\sqrt{d}.$$

Therefore

$$\frac{\mathbf{P}(x \text{ comes from } G_j)}{\mathbf{P}(x \text{ comes from } G_i)} \;\leq\; \frac{w_j e^{-(c_{ij}^*-\rho)^2 d/2}}{w_i e^{-(\mathrm{E}^*\rho)^2 d/2}} \;\leq\; \frac{w_j}{w_{min}} e^{-3c_{ij}^{*2}d/8}$$

since $\rho \leq \frac{1}{16\mathrm{E}^*}$ while $c_{ij}^* \geq \frac{1}{2}$. Under the assumption $d \geq 16 \ln \frac{10}{\rho w_{min}} + \frac{8}{e}$, and using the simple inequality $\ln x \leq \frac{x}{e}$ (for $x \geq 0$), we find that $\mathbf{P}(x \text{ comes from } G_j)$ is at most $\frac{w_j\rho}{10c_{ij}^*}e^{-3c_{ij}^*/2}$. We can now use a Chernoff inequality (Appendix A.1) to bound the chance that l_{ij} is very much more than l times this probability. Specifically, since by assumption $l \geq \frac{30}{w_{min}\rho}\ln\frac{k^2}{\delta}$,

$$\mathbf{P}\left(\frac{l_{ij}}{l} > \frac{w_j\rho}{10c_{ij}^*}\right) \leq e^{-lw_j\rho/30} \leq \frac{\delta}{k^2},$$

as claimed. ∎

4 High-dimensional reconstruction

4.1 Summary

Theorem 14 *The third phase of the algorithm takes as input the sets $S_i^* \subset \mathbb{R}^d$ of size l, and returns the averages $\widehat{\mu}_i$ of the corresponding sets $S_i \subset \mathbb{R}^n$. If the following conditions are satisfied:*

- *$S_i^* \subset B(\mu_i^*; \rho\sigma_{max}^*\sqrt{d})$ for all i, for some $\rho \leq \frac{1}{4}$,*
- *the proportion of S_i^* from a different Gaussian $G_j, j \neq i$, is $\frac{l_{ij}}{l} \leq \frac{w_j\rho}{10c_{ij}^*}$,*
- *$n \geq 16 \log \frac{k}{\delta}$, and*
- *$d \leq \frac{n}{2}, c_{ij}^* \geq \frac{1}{2}c_{ij} \geq \frac{1}{2}$,*

then with probability at least $1 - \delta$, for each $1 \leq i \leq k$,

$$\|\widehat{\mu}_i - \mu_i\| \leq \left(\frac{\rho\mathrm{E}^*}{2} + \sqrt{\frac{2}{l}}\right)\sigma_{max}\sqrt{n}.$$

In particular, if $\rho \leq \frac{1}{16\mathrm{E}^*}$, choosing $l \geq 2048$ will make this error at most $\frac{1}{16}\sigma_{max}\sqrt{n}$.

Proof. This is shown over the course of Lemmas 15 to 17. ∎

4.2 Details

So far we have data points S_i^* near each of the projected means. Let S_i correspond to these same points in the original space. The high-dimensional estimate $\widehat{\mu}_i$ in $\mathbb{R}^n$ is the mean of S_i.

The random projection from $\mathbb{R}^n$ to $\mathbb{R}^d$ can be thought of as a composition of two linear transformations: a random rotation in $\mathbb{R}^n$ followed by a projection onto the first d coordinates. Since rotations preserve L_2 distance, we can assume, for the purpose of bounding the L_2 accuracy of our high-dimensional center-estimates, that the random projection consists solely of a mapping onto the first d coordinates. We will write high-dimensional points in the form $(x, y) \in \mathbb{R}^d \times \mathbb{R}^{n-d}$, and will assume that each such point is projected down to x. We have already bounded the error on the first portion. How do we deal with the rest? We start with an important

Definition T_{ij} = points in S_i drawn from the j^{th} Gaussian G_j. As before, let $l_{ij} = |T_{ij}|$.

We have seen that S_i is relatively uncontaminated by points from other Gaussians, that is, $\sum_{j \neq i} l_{ij}$ is small. Those points which do come from G_i ought to (we hope) average out to something near its mean μ_i. The problem is that the x coordinates could be highly correlated with the y coordinates (depending upon the nature of Σ), and thus a small, unavoidable error in $\widehat{\mu}_i^*$ might potentially cause the set T_{ii} to lie far from μ_i in $\mathbb{R}^n$. To dismiss this possibility we need a bit of matrix analysis.

Write covariance matrix Σ in the form $\begin{pmatrix} \Sigma_{xx} & \Sigma_{xy} \\ \Sigma_{yx} & \Sigma_{yy} \end{pmatrix}$, with $\Sigma_{xx} = \Sigma^*$ being the covariance matrix of the projected Gaussians. Our operational assumption throughout has been that Σ has finite eccentricity and is therefore positive definite. Now, what is the relationship between the x and y components of points drawn from Gaussians with covariance Σ?

Fact If a point drawn randomly from $N(0, \Sigma)$ has x as its first d coordinates, then its last $n - d$ coordinates have the induced (conditional) distribution $N(Ax, C)$, where $A = \Sigma_{yx}\Sigma_{xx}^{-1}$ and $C = \Sigma_{yy} - \Sigma_{yx}\Sigma_{xx}^{-1}\Sigma_{xy}$. This well-known result can be found, for instance, in Lauritzen's (1996) book on graphical models.

We will need to tackle the question: for a point (x, y) drawn from $N(0, \Sigma)$, what is the distribution of $\|y\|$ given $\|x\|$? In order to answer this, we need to study the matrix A a bit more carefully.

Lemma 15 $\|Ax\| \leq \sigma_{max}\|x\|_{\Sigma^*}\sqrt{n/d}$ *for any* $x \in \mathbb{R}^d$.

Proof. $A = \Sigma_{yx}\Sigma_{xx}^{-1}$ is an $(n-d)\times d$ matrix; divide it into $\frac{n}{d}-1$ square matrices $B_1, \ldots, B_{n/d-1}$ by taking d rows at a time. Fix attention on one such B_i. The rows of B_i correspond to some d consecutive coordinates of x; call these coordinates z. Then we can write $B_i = \Sigma_{zx}\Sigma_{xx}^{-1}$. It is well-known – see, for instance, the textbook by Horn and Johnson (1985), or consider the inverse of the $2d \times 2d$ positive definite covariance matrix of (z,x) – that $(\Sigma_{xx} - \Sigma_{xz}\Sigma_{zz}^{-1}\Sigma_{zx})$ is positive definite. Therefore, for any $u \in \mathbb{R}^d$, $u^T\Sigma_{xx}u > u^T\Sigma_{xz}\Sigma_{zz}^{-1}\Sigma_{zx}u$, and by choosing $u = \Sigma_{xx}^{-1}v$, we find

$$\|v\|_{\Sigma^*}^2 \;=\; u^T\Sigma_{xx}u \;>\; v^T B_i^T\Sigma_{zz}^{-1}B_i v \;\geq\; \|B_i v\|^2\lambda_{min}(\Sigma_{zz}^{-1}) \;\geq\; \frac{\|B_i v\|^2}{\sigma_{max}^2}.$$

Therefore $\|B_i v\| \leq \sigma_{max}\|v\|_{\Sigma^*}$. The pieces now come neatly together,

$$\|Ax\|^2 \;=\; \|B_1 x\|^2 + \cdots + \|B_{n/d-1}x\|^2 \;\leq\; (\tfrac{n}{d}-1)\sigma_{max}^2\|x\|_{\Sigma^*}^2,$$

and the lemma is proved. ▮

For the analysis we are pretending that we know the x coordinates of all the data points but that their y coordinates have not yet been chosen. We have seen that points in T_{ij} roughly have conditional distribution $\mu_j + (\widehat{\mu}_i^* - \mu_j^*, N(A(\widehat{\mu}_i^* - \mu_j^*), C))$. What bounds can be given for the average of these points? In particular we would like to bound the deviation of mean(T_{ij}) from μ_j and thereby from μ_i.

Lemma 16 *For any* $j \neq i$,

$$mean(T_{ij}) \overset{d}{=} \mu_j + (X, AX + C^{1/2}N(0, \tfrac{1}{l_{ij}}I_{n-d})),$$

where $X \in \mathbb{R}^d$ *is a random variable with* $\|X\| \leq \|\mu_i^* - \mu_j^*\| + \rho\sigma_{max}^*\sqrt{d}$.

Proof. For the sake of convenience translate the Gaussians and data points by $-\mu_j$. This centers the j^{th} Gaussian G_j at the origin and centers G_i at $\mu_i - \mu_j$. The translated S_i^* now lies within $B(\mu_i^* - \mu_j^*; \rho\sigma_{max}^*\sqrt{d})$. Recall that T_{ij} consists of those points in S_i which come from Gaussian G_j. Therefore, for our purposes we can pretend that each point $(X_a, Y_a) \in T_{ij}$, in the translated space, is generated in the following fashion:

- Pick $X_a \in B(\mu_i^* - \mu_j^*; \rho\sigma_{max}^*\sqrt{d}) \subset \mathbb{R}^d$ in some unknown manner.
- Choose $Y_a \overset{d}{=} N(AX_a, C)$.

In this manner we choose $l_{ij} = |T_{ij}|$ points, with mean value some (X, Y). The range of the X_a coordinates constrains $\|X\|$ to be at most $\|\mu_i^* - \mu_j^*\| + \rho\sigma_{max}^*\sqrt{d}$. To understand the distribution of Y_a, we notice $(Y_a - AX_a) \overset{d}{=} N(0, C) \overset{d}{=} C^{1/2}N(0, I_{n-d})$, and taking averages, $Y \overset{d}{=} AX + C^{1/2}N(0, \frac{1}{l_{ij}}I_{n-d})$. ▮

Armed with this result we can prove the correctness of this phase of the algorithm.

Lemma 17 *Under the conditions of Theorem 14, with probability at least* $1-\delta$, *for all* $1 \le i \le k$, $\|\widehat{\mu}_i - \mu_i\| \le \left(\frac{\rho \mathrm{E}^*}{2} + \sqrt{\frac{2}{l}}\right)\sigma_{max}\sqrt{n}$.

Proof. Fix any i. The normed difference between μ_i and the mean of S_i, which we hope is close to zero, is given by

$$\begin{aligned}
&\|\text{mean}(S_i) - \mu_i\| \\
&\le \left\|\sum_{j=1}^{k}(\text{mean}(T_{ij}) - \mu_j)\frac{l_{ij}}{l}\right\| + \sum_{j\neq i}\|\mu_j - \mu_i\|\frac{l_{ij}}{l} \\
&\le \sum_{j=1}^{k}\frac{l_{ij}}{l}\left\{\left(\|\mu_i^* - \mu_j^*\| + \rho\sigma_{max}^*\sqrt{d}\right)\left(1 + \frac{\sigma_{max}}{\sigma_{min}^*}\sqrt{\frac{n}{d}}\right)\right\} + \|C^{1/2}N_l\| + \sum_{j\neq i}\|\mu_j - \mu_i\|\frac{l_{ij}}{l} \\
&\le \sum_{j=1}^{k}\frac{l_{ij}}{l}\left(5c_{ij}^*\mathrm{E}^*\sigma_{max}\sqrt{n}\right) + \|C^{1/2}N_l\|
\end{aligned}$$

where N_l has distribution $N(0, \frac{1}{l}I_{n-d})$ and the second inequality uses Lemmas 15 and 16. It remains to bound these last two terms.

(a) Since $C = \Sigma_{yy} - \Sigma_{yx}\Sigma_{xx}^{-1}\Sigma_{xy}$ and each of these two right-hand terms is positive semidefinite, $\lambda_{max}(C) \le \lambda_{max}(\Sigma_{yy}) \le \sigma_{max}^2$ and therefore $\|C^{1/2}N_l\| \le \sigma_{max}\|N_l\|$. Bounding $\|N_l\|$ is routine (Lemma A.3). In particular, if $d \le \frac{1}{2}n$,

$$\mathbf{P}(\|N_l\| > \sqrt{\tfrac{2n}{l}}) \;\le\; e^{-n/16} \;\le\; \tfrac{\delta}{k}.$$

(b) By assumption, $l_{ij} \le \frac{lw_j\rho}{10c_{ij}^*}$. Applying this bound immediately yields the lemma. ▮

5 Consolidation in the high-dimensional space

The first three phases of the algorithm obtain very rough estimates of the high-dimensional centers. These are enough to cluster the data almost perfectly, and with this clustering we can obtain sharp estimates of the means, covariances, and mixing weights.

Associate each data point in S with its closest center-estimate $\widehat{\mu}_i$. It can be shown that the expected proportion of misclassified points in this *hard clustering* is small, roughly

$O(ke^{-\Omega(n)})$. The proof is not difficult but requires some care because of the dependence between the data and the estimated centers.

Lemma 18 *Suppose the true mixture $\{(w_i, \mu_i, \Sigma)\}$ is 1-separated, and that the estimated centers $\widehat{\mu}_i$ satisfy*

$$\|\widehat{\mu}_i - \mu_i\| \leq \tfrac{1}{16}\sigma_{max}\sqrt{n}.$$

In the hard clustering, a point x drawn from $N(\mu_i, \Sigma)$ will not be misclassified as belonging to $N(\mu_j, \Sigma)$, $j \neq i$, if
(a) $\|x - \mu_i\| \leq \frac{5}{4}c_{ij}\sigma_{max}\sqrt{n}$ and
(b) $(x - \mu_i) \cdot \frac{\mu_j - \mu_i}{\|\mu_i - \mu_j\|} \leq \frac{1}{6}c_{ij}\sigma_{max}\sqrt{n}$.

Proof. Pick any x from $N(\mu_i, \Sigma)$ and any $j \neq i$. We will show that under the conditions above, $\|x - \widehat{\mu}_i\| < \|x - \widehat{\mu}_j\|$, and thus x will not be misclassified.

We start by rewriting $\|x - \widehat{\mu}_j\|^2 - \|x - \widehat{\mu}_i\|^2$, which we hope is positive, as $\|\widehat{\mu}_i - \widehat{\mu}_j\|^2 - 2(x - \widehat{\mu}_i) \cdot (\widehat{\mu}_j - \widehat{\mu}_i)$. The quantity $\|\widehat{\mu}_j - \widehat{\mu}_i\|$ differs from $\|\mu_j - \mu_i\|$ by at most $\frac{1}{8}\sigma_{max}\sqrt{n}$, and thus is at least $\frac{7}{8}c_{ij}\sigma_{max}\sqrt{n}$. The second term, meanwhile, depends upon the interaction of the data and the estimated centers, and so we would like to rewrite it in terms of the data and the true centers.

$$\begin{aligned}
(x - \widehat{\mu}_i) \cdot (\widehat{\mu}_j - \widehat{\mu}_i) &= \\
& (x - \mu_i) \cdot (\mu_j - \mu_i) - (x - \mu_i) \cdot ((\mu_j - \widehat{\mu}_j) - (\mu_i - \widehat{\mu}_i)) - (\widehat{\mu}_i - \mu_i) \cdot (\widehat{\mu}_j - \widehat{\mu}_i) \\
&\leq (x - \mu_i) \cdot (\mu_j - \mu_i) + \|x - \mu_i\| \cdot (\|\mu_j - \widehat{\mu}_j\| + \|\mu_i - \widehat{\mu}_i\|) + \|\widehat{\mu}_i - \mu_i\| \cdot \|\widehat{\mu}_j - \widehat{\mu}_i\| \\
&\leq \frac{1}{6}c_{ij}\|\mu_j - \mu_i\|\sigma_{max}\sqrt{n} + \frac{5}{4}c_{ij}\sigma_{max}\sqrt{n} \cdot \frac{1}{8}\sigma_{max}\sqrt{n} + \frac{1}{16}\sigma_{max}\sqrt{n} \cdot \|\widehat{\mu}_j - \widehat{\mu}_i\| \\
&= \frac{1}{6}c_{ij}^2\sigma_{max}^2 n + \frac{5}{32}c_{ij}\sigma_{max}^2 n + \frac{1}{16}\sigma_{max}\sqrt{n} \cdot \|\widehat{\mu}_j - \widehat{\mu}_i\| \\
&< \frac{1}{2}\|\widehat{\mu}_j - \widehat{\mu}_i\|^2.
\end{aligned}$$

Therefore $\|x - \widehat{\mu}_j\|^2 - \|x - \widehat{\mu}_i\|^2 > 0$, as we had hoped. ∎

This lemma makes it easy to show that the hard clustering is close to optimal.

Lemma 19 *Fix any two indices $i \neq j$, and draw X randomly from $N(\mu_i, \Sigma)$. Then the probability that X violates either condition (a) or (b) of the previous lemma is at most $e^{-c_{ij}^2 n/20} + \frac{6}{c_{ij}\sqrt{2\pi n}}e^{-c_{ij}^2 n/72}$, assuming as before that $c_{ij} \geq 1$.*

Proof. By Lemma A.3,

$$\mathbf{P}(\|X-\mu_i\| > \tfrac{5}{4}c_{ij}\sigma_{max}\sqrt{n}) \;\leq\; e^{-c_{ij}^2 n/20}.$$

For any unit direction γ, the random variable $(X-\mu_i)\cdot\gamma$ has distribution $N(0,\gamma^T\Sigma\gamma)$, and this variance $\gamma^T\Sigma\gamma$ is at most σ_{max}^2. Therefore, for any $j \neq i$,

$$\mathbf{P}((X-\mu_i)\cdot \tfrac{\mu_j-\mu_i}{\|\mu_j-\mu_i\|} > \tfrac{1}{6}c_{ij}\sigma_{max}\sqrt{n}) \;\leq\; \mathbf{P}(Z > \tfrac{1}{6}c_{ij}\sqrt{n}),$$

where Z is a standard normal random variable. This last probability can routinely be bounded (Durrett, 1996, p.7) to complete the lemma. ▮

This more or less completes the proof. The chance that a particular point is misclassified in phase four is at most k times this value, which is miniscule if n is large. Specifically, if $m' = \frac{2}{\epsilon^2 w_{min}} \ln\frac{k}{\delta}$ points are chosen randomly from the data set, then with probability at least $1-\delta$, there will be at least $\frac{2}{\epsilon^2}$ points from each Gaussian. The chance that none of these points is misclassified is very high, at least $1-\delta$ if $n \geq \Omega(d + \log\frac{1}{\epsilon})$. And, $\frac{2}{\epsilon^2}$ points from a Gaussian are enough to specify its mean within L_2 distance $\epsilon\sigma_{max}\sqrt{n}$, with probability at least $1-e^{-n/8}$. In other words, by computing statistics of the points in each "hard" cluster, we can expect to get good estimates of the means, as well the mixing weights and covariance matrix.

The final phase of the algorithm is similar to a single 'M' step of EM (although it uses a hard clustering like k-means) while phases one, two, and three are devoted to finding a reasonable initial solution. In particular, the accuracy required of the first three phases, which we have been calling ρ, is unrelated to the accuracy the user demands.

Well-separated spherical Gaussians in very high dimension are easy to learn. Data drawn from them can be clustered just by looking at interpoint distances. However, spherical clusters are unlikely to occur naturally in data because different attributes are typically measured along different scales. Random projection helps to make arbitrary Gaussians look spherical. We will also see that it makes a wide range of different distributions look Gaussian, in the sense of the "weak Gaussian assumption". These beneficial effects get more pronounced as the projected dimension is lowered.

Our algorithm therefore lowers the dimension as much as possible, in the hope of making the data look almost spherical-Gaussian. This dimension is so low that efficient clustering can be managed but is no longer trivial.

6 Shortcomings

The least satisfactory part of this algorithm is the subroutine which estimates centers in low dimension. Some of its problems are fairly minor. For instance, the need to assess distances between all pairs of points, unthinkable for very large data sets, can be made tolerable by instead computing distances from all points to a small random sample of the data. However, there are two particular flaws which deserve careful attention. First, the "weak Gaussian assumption" might not be quite weak enough. The practical applicability of the algorithm would be more believable if it could be shown to robustly handle data which at some time satisfied the weak assumption but which has since been corrupted by a small amount of adversarial noise. The second inadequacy is the requirement that all clusters have approximately the same radius. Rectifying this would be a significant advance in current clustering technology.

Open Problem 1 Is there a simple algorithm which correctly learns the centers of a mixture of spherical Gaussians with widely differing radii?

A seemingly easier question is what to do when the number of clusters is unknown. The algorithm of this chapter will behave predictably if given the wrong value of k. Therefore, it should be possible to test various values, $k = 1, 2, \ldots$, but there must be a cleaner approach.

Open Problem 2 Can our algorithm be modified to accommodate situations in which the exact number of clusters is unknown but a loose upper bound can reliably be guessed?

7 More general mixture models

The recent machine learning literature contains some interesting work on clusters whose empirical distributions are non-Gaussian in the sense of being discrete or "fat-tailed".

Kearns, Mansour, Ron, Rubinfeld, Schapire and Sellie (1994) have considered a discrete analogue of spherical Gaussians which they call *Hamming balls*. These are specified by a center $\mu \in \{0,1\}^n$ and a corruption probability $p \in (0,1)$. The distribution is then $(B(p) \times B(p) \times \cdots \times B(p)) \oplus \mu$, where $B(p)$ denotes a Bernoulli random variable with bias p (it has value one with probability p and value zero otherwise) and $\oplus$ is exclusive-or. Given data from a mixture of k such balls, the authors show how to find a mixture of $O(k \log k)$

balls which comes close in distribution to the original mixture. Freund and Mansour (1999) consider the related model of product distributions over $\{0,1\}^n$, that is, distributions of the form $B(p_1) \times \cdots \times B(p_n)$ for some $p_1, \ldots, p_n \in (0,1)$. They present a simple algorithm for estimating the parameters of a mixture of two such distributions, in a learning framework similar to ours.

Discrete product distributions seem similar enough in spirit to Gaussians that it should be possible to extend the techniques of this chapter to accommodate them. *Fat-tailed* distributions, on the other hand, can contain strong dependencies between different coordinates. Rather than attempt a precise definition we shall, roughly speaking, describe as fat-tailed any distribution over vectors $X \in \mathbb{R}^n$ for which $\|X\|^2$ falls off significantly more slowly than in the Gaussian case, in other words, either $\mathbf{E}\|X\|^2$ does not exist, or if it does, $\mathbf{P}(\|X\|^2 > c\mathbf{E}\|X\|^2) = e^{-o(c^2 n)}$.

Example 1 One generic way of constructing such a distribution is from a stable law of exponent less than two. The standard Cauchy density in $\mathbb{R}^1$,

$$\gamma_t(x) = \frac{1}{\pi} \cdot \frac{t}{t^2 + x^2},$$

approaches zero with such hesitation that its expectation does not exist. Therefore, a high-dimensional distribution with Cauchy coordinates will be fat-tailed. ▮

Example 2 It might be of greater practical interest to consider distributions whose coordinates are bounded, perhaps even Boolean. In such cases, a fat tail is the result of dependencies between the various attributes. For example, consider a distribution D over vectors $X = (X_1, \ldots, X_n) \in \{0,1\}^n$ which is constructed by first choosing $Y = (Y_1, \ldots, Y_{\sqrt{n}})$ uniformly at random from $\{0,1\}^{\sqrt{n}}$ and then setting $X_i = Y_{\lceil i/\sqrt{n} \rceil}$. Therefore X_1 is independent of $X_{\sqrt{n}+1}$ but is always equal to $X_2, \ldots, X_{\sqrt{n}}$. The squared length $\|X\|^2$ can be written as $\sqrt{n}\|Y\|^2$ and has expectation $\mathbf{E}\|X\|^2 = \frac{1}{2}n$. What is the chance of a large deviation from this expected value?

$$\mathbf{P}(\|X\|^2 \geq \tfrac{3}{4}n) = \mathbf{P}(Y_1 + \cdots + Y_{\sqrt{n}} \geq \tfrac{3}{4}\sqrt{n}) = e^{\theta(\sqrt{n})},$$

and so the distribution is fat-tailed. ▮

Basu and Micchelli (1998) report experiments which attempt to model speech data using mixtures of fat-tailed densities

$$p(x) \propto \exp\left(-c((x-\mu)^T \Sigma^{-1} (x-\mu))^\alpha\right)$$

where $\alpha < 1$ (the case $\alpha = 1$ produces a Gaussian). It turns out that this particular family of distributions can be accommodated by an algorithm similar to EM. In reading this and other such highly specialized results, the main question that comes to mind is whether there exists a unified way of dealing with a heterogeneous collection of distributions, some of which may be discrete or fat-tailed?

7.1 Some consequences of the central limit theorem

A random point $X = (X_1, \ldots, X_n)$ drawn uniformly from the n-dimensional cube $\{-1, 1\}^n$ has i.i.d. coordinates with mean zero and variance one. The central limit theorem can be applied to the projection of X in the direction $\gamma = \frac{1}{\sqrt{n}}(1, 1, \ldots, 1)$,

$$\gamma \cdot X \quad = \quad \frac{1}{\sqrt{n}}(X_1 + \cdots + X_n) \quad \xrightarrow{d} \quad N(0, 1)$$

as n increases to infinity. That is to say, the projection of this discrete distribution in the particular direction γ looks like a Gaussian. In fact, this will hold for almost any randomly chosen direction.

A celebrated paper of Diaconis and Freedman (1984) studies this phenomenon in detail. They find a large family of high-dimensional distributions almost all of whose one-dimensional projections are close to Gaussian. This is terrible news for anyone who is trying to "visualize" a cluster by projecting it onto random lines, but for us it immediately suggests a simple and unexpected methodology for learning a large class of mixture models: project the data into a random low-dimensional subspace in the hope that this will make the clusters more Gaussian, learn these clusters under a Gaussian assumption, and then perform some kind of high-dimensional reconstruction. Before we get carried away, we should take a closer look at the fine print. First we review a classical result of Berry and Esséen (see Feller, 1966, or Petrov, 1995).

Theorem 20 (Rate of convergence in the central limit theorem) *Let $X_1, X_2, \ldots$ be a sequence of independent random variables with $\mathbf{E}X_i = 0$ and $\mathbf{E}X_i^2 = \sigma_i^2$. Write $s_n = \sqrt{\sigma_1^2 + \cdots + \sigma_n^2}$. Then there is a universal constant $C > 0$ such that for all n, the distribution F_n of the normalized sum $\frac{1}{s_n}(X_1 + \cdots + X_n)$ is close to the distribution Φ of the standard normal $N(0, 1)$ in that*

$$\sup_{-\infty < a < b < \infty} |F_n(a, b) - \Phi(a, b)| \ \leq \ C s_n^{-3} \sum_{i=1}^{n} \mathbf{E}|X_i|^3.$$

Theorem 21 (Diaconis and Freedman, 1984) *Suppose vectors $X \in \mathbb{R}^n$ have i.i.d. coordinates with mean zero, variance one and third absolute moment ρ_3. Then for almost all unit vectors γ, the distribution F_n of $\gamma \cdot X$ is close to normal in the sense that*

$$\sup_{-\infty<a<b<\infty} |F_n(a,b) - \Phi(a,b)| \;\leq\; 2C\rho_3 n^{-1/2}.$$

Proof. Fix a direction $\gamma \in \mathbb{R}^n$. Applying the Berry-Esséen theorem to the independent random variables $\gamma_1 X_1, \ldots, \gamma_n X_n$ reveals that the distribution $F_n \overset{d}{=} \gamma \cdot X$ satisfies

$$\sup_{-\infty<a<b<\infty} |F_n(a,b) - \Phi(a,b)| \;\leq\; C\rho_3 \sum_{i=1}^n |\gamma_i|^3.$$

A randomly chosen γ from the unit sphere in $\mathbb{R}^n$ has distribution

$$\gamma \;\overset{d}{=}\; \frac{(Z_1, Z_2, \ldots, Z_n)}{\sqrt{Z_1^2 + \cdots + Z_n^2}},$$

where the Z_i are independent standard normals. This means in particular that

$$\sum_{i=1}^n |\gamma_i|^3 \;\overset{d}{=}\; \frac{\sum_{i=1}^n |Z_i|^3}{(\sum_{i=1}^n Z_i^2)^{3/2}} \;=\; n^{-1/2} \cdot \frac{(1/n)\sum_{i=1}^n |Z_i|^3}{((1/n)\sum_{i=1}^n Z_i^2)^{3/2}}.$$

Since $\mathbf{E}Z_i^2 = 1$ and $\mathbf{E}|Z_i|^3 = 4/\sqrt{2\pi} < 2$, it follows by applying large deviation bounds to the numerator and denominator that the chance (over choice of γ) that the final quotient is more than two is $e^{-\Omega(n)}$. ▮

Similar statements can be made when the coordinates of the vector X are independent but not identically distributed. Presumably this closeness to normality continues to hold when the random projection is to a d-dimensional space rather than a line, for small d. In this more general case, we are interested in the discrepancy on spherical regions, for the weak Gaussian assumption.

Open Problem 3 Do these results still hold for projections onto random subspaces of (small) dimension d? Let $\mathbf{B}$ denote the class of all spheres in $\mathbb{R}^d$. By how much does the projected distribution differ from normal on sets in $\mathbf{B}$? How does this difference scale as a function of d?

7.2 Distributions with non-independent coordinates

The distribution D of Example 2 contains heavy dependencies between some of the coordinates. Recall that a vector $X = (X_1, \ldots, X_n)$ is drawn from D by first selecting

$Y = (Y_1, \ldots, Y_{\sqrt{n}})$ uniformly at random from $\{0,1\}^{\sqrt{n}}$ and then setting $X_i = Y_{\lceil i/\sqrt{n} \rceil}$. What can we expect of its random projection?

For a unit direction γ,

$$\gamma \cdot X \overset{d}{=} \alpha_1 Y_1 + \cdots + \alpha_{\sqrt{n}} Y_{\sqrt{n}},$$

where $\alpha_i = \gamma_{(i-1)\sqrt{n}+1} + \cdots + \gamma_{i\sqrt{n}}$. If γ is chosen randomly then $(\alpha_1, \ldots, \alpha_{\sqrt{n}})$, when normalized to unit length, will itself be a random unit vector; therefore the results for the i.i.d. case will apply once more, except this time the distance from normality will be proportional to $n^{-1/4}$ rather than $n^{-1/2}$.

Open Problem 4 If a distribution over vectors $X \in \{0,1\}^n$ has entropy at least $n^{\Omega(1)}$, it is true that for most random unit directions γ, the projected distribution $F_n = \gamma \cdot X$, suitably normalized, satisfies $\sup_{a<b} |F_n(a,b) - \Phi(a,b)| \leq n^{-\Omega(1)}$?

Example 3 This one is due to David Zuckerman. Consider a distribution D' on $\{0,1\}^n$ which is uniform over

$$T = \{\text{strings with exactly } \tfrac{1}{3}n \text{ ones}\} \cup \{\text{strings with exactly } \tfrac{1}{3}n \text{ zeros}\}$$

and is zero elsewhere. D' has high entropy $\theta(n)$ because the size of T is exponential in n. However, a vector $X = (X_1, \ldots, X_n)$ drawn randomly from D' will not satisfy the law of large numbers because $\frac{1}{n}(X_1 + \cdots + X_n)$ cannot be close to its expected value $\frac{1}{2}$; it will always be either $\frac{1}{3}$ or $\frac{2}{3}$. This means that the distribution of $(X_1 + \cdots + X_n)/\sqrt{n}$ is far from Gaussian. The open problem expresses the hope that, nevertheless, for *most* directions γ, the projection $\gamma \cdot X$ will look Gaussian. ▮

Chapter IV

Probabilistic nets

1 Background

Directed probabilistic nets can express distributions far more complicated than Gaussian mixtures. They pay for this representational power by residing in a thick swamp of computational intractability. However there are a few positive algorithmic results (Edmonds, 1967; Chow and Liu, 1968; Pearl, 1988; Lauritzen and Spiegelhalter, 1988; Jaakkola and Jordan, 1999), quite modest in scope, and to which the current popularity of probabilistic nets can directly be traced. Before reviewing these we as usual need to agree upon a barrage of notation.

A *directed probabilistic net* over the random variables $X_1, \ldots, X_n$ assigns each a node and imposes a directed acyclic graph structure upon them. Let $\Pi_i \subseteq \{X_1, \ldots, X_n\}$ denote the *parents* of node X_i, that is, the nodes which point to X_i. Then X_i is reciprocally called a *child* of any member of Π_i. The *indegree* of X_i is the number of parents it has, $|\Pi_i|$. The acyclic structure makes it inevitable that at least one node will have no parent. Such nodes are called *sources*. A distribution over the net is specified via conditional probability distributions $\mathbf{P}(X_i|\Pi_i)$ whose product is the joint distribution on $(X_1, \ldots, X_n)$.

A maximal structure has nodes with indegrees $1, 2, \ldots, n-1$ (Figure 1, left-hand side) and can represent all distributions over the random variables. Often, however, a distribution exhibits various approximate conditional independencies which can be exploited to reduce the complexity of the structure graph. A tension is thus created, in which model size is reduced at the expense of accuracy.

1.1 Conditional probability functions

The conditional probability distributions can be specified explicitly in the form of tables if the random variables have a finite number of possible values. These are predictably called *conditional probability tables* (CPTs). If all random variables are binary-valued, then a node X_i with k parents Π_i requires a CPT of size 2^k. Each entry in the table denotes the probability that X_i is one given a particular setting of Π_i.

Restrictive alternatives to CPTs are frequently used in the interest of saving space. The most popular of these appears to be the *noisy-OR gate.* For a node $X_i = X$ with parents $\Pi_i = \{Z_1, \ldots, Z_k\}$, all binary-valued, a noisy-OR conditional probability distribution is specified by $k + 1$ numbers $\Delta, p_1, \ldots, p_n \in [0, 1]$, where

$$\mathbf{P}(X = 0|Z_1, \ldots, Z_k) = (1 - \Delta) \prod_{i=1}^{k} (1 - p_i)^{Z_i}.$$

This is meant to be a probabilistic OR gate. Raising any of the parent values $Z_1, \ldots, Z_k$ can only increase the chance that X is one, so the noisy-OR might be inappropriate if X is not an increasing function of its parents. Writing $w_i = -\ln(1 - p_i)$ and $w_0 = -\ln(1 - \Delta)$, we can express the distribution of X in a clearer pseudo-linear form,

$$\mathbf{P}(X = 0|Z_1, \ldots, Z_k) = \exp\left(-w_0 - \sum_{i=1}^{k} w_i Z_i\right).$$

If the variables can take on a continuous range of values and a decision is made not to discretize them, the most common choice of conditional probability function is the *linear-Gaussian.* For X with parents $Z_1, \ldots, Z_k$ as before, this is specified by a vector $v \in \mathbb{R}^k$ and a noise parameter ϵ:

$$X \overset{d}{=} v \cdot (Z_1, \ldots, Z_k) + N(0, \epsilon^2).$$

In other words, X is assumed to be a linear function of its parents, with some independent Gaussian noise added in to make whole thing look stochastic.

1.2 Inference

Probabilistic nets can be used to address arbitrary conditional probability queries about the represented random variables, to evaluate probabilities such as $\mathbf{P}(X_1 = 1|X_2 = 0, X_3 = 1, X_4 = 0)$. The QMR-DT net described in the introduction (Figure I.2) would not be especially useful if one could not ask “what is the chance the patient has an ulcer given that he has the following symptoms: . . . ?" The process of answering such queries is called *inference* in the artificial intelligence literature.

Since probabilistic nets are a generalization of Boolean circuits, any inference procedure can be used to solve NP-complete problems like "does this Boolean circuit ever produce the result zero?" It follows immediately that inference is NP-hard to perform even approximately, to within any multiplicative factor (Cooper, 1990). Dagum and Luby (1993) show that inference is also NP-hard to perform within a constant additive accuracy, even for probabilistic nets in which all indegrees and outdegrees are small, say three or less.

These hardness results very seriously call into question the practicality of probabilistic nets, and make it critically important to find subclasses of nets for which efficient, possibly approximate, inference is possible. The most notable such class is that of directed trees, called *polytrees* in some of the literature, which permit exact inference to be performed quickly by dynamic programming, or by a convenient message-passing formalism (Pearl, 1988).

There are a few other positive results for inference. The *junction tree* algorithm (Lauritzen, 1996) works efficiently for a small class of graphs whose characterization is somewhat intricate but which includes polytrees. For nets composed of layers of noisy-OR gates (or various other classes of conditional probability functions), variational approximations can be used as part of an inference heuristic (Jaakkola and Jordan, 1999). The quality of these approximations differs between instances, but in each case the final result is accompanied by nontrivial bounds on its error. This technique has worked impressively on the QMR-DT net.

2 A maximum-likelihood learning model

2.1 Entropy

The *entropy* of a random variable X with discrete distribution D is defined as

$$H(X) = H(D) = \sum_x D(x) \log_2 \frac{1}{D(x)}.$$

This value is best thought of as a measure of the inherent randomness content of X, scaled so that a fair coin has unit entropy. A biased coin has a slightly less random outcome and therefore has entropy less than one. A deterministic event has zero entropy.

If X, Y are discrete random variables with distribution D, the *conditional entropy* of X given Y is the average amount of randomness left in X once Y is known, and is defined as

$$H(X|Y) = \sum_{y} D(Y = y) H(X|Y = y).$$

This definition of randomness has various satisfying properties (Cover and Thomas, 1991).

- $H(X) \geq 0$.
- $H(X, Y) = H(X) + H(Y|X)$.
- $H(X, Y) \leq H(X) + H(Y)$, with equality if and only if X and Y are independent. Therefore $H(Y|X) \leq H(Y)$.
- If X can only take values in some finite set S then $H(X) \leq \log_2 |S|$, with equality if and only if X is uniformly distributed over S.

2.2 Maximizing likelihood

Suppose a collection of m data points is obtained from some arbitrary target distribution D over $(X_1, \ldots, X_n)$, and a probabilistic net is sought which fits this data well. The log-likelihood of a particular model θ is

$$\mathrm{LL}(\theta) = \frac{1}{m} \sum_{i=1}^{m} \log \left(\text{Probability that } \theta \text{ assigns to the } i^{th} \text{ data point}\right).$$

Let Π_i^θ be the parents that θ chooses for X_i. The maximum-likelihood choice is to set the conditional probabilities $\mathbf{P}(X_i = x_i | \Pi_i^\theta = \pi_i)$ to their empirically observed values. The log-likelihood then becomes

$$\mathrm{LL}(\theta) = -\sum_{j=1}^{n} H(X_i | \Pi_i^\theta),$$

where $H(X_i|\Pi_i^\theta)$ is computed with respect to the empirical distribution consisting of the m samples.

The question "how many samples are needed so that a good fit to the data reflects a good fit to D?" has been considered elsewhere (Dasgupta, 1997; Höffgen, 1993), and so we will not dwell upon sample complexity issues here. Henceforth we blur the distinction between D and the empirically observed distribution.

Maximizing the log-likelihood now corresponds to choosing a directed graph G so as to minimize the cost function

$$\mathrm{cost}(G) = \sum_{j=1}^{n} H(X_i|\Pi_i^G),$$

where the conditional entropy is with respect to D and Π_i^G denotes the parents that G assigns to node X_i. The target distribution has an inherent randomness content $H(D)$ and there is no graph whose cost is less than this. The most naive structure, which has n nodes and no edges, will have cost $\sum_i H(X_i)$, and there is similarly no graph of higher cost.

Choosing G to be maximally connected will ensure that the target distribution can be modelled perfectly and that the minimum possible cost, $H(D)$, is achieved. However the corresponding CPTs are of size exponential in n and so this is hardly a compact representation of D. The size of one of these models is directly related to the number of parents that nodes are allowed to have. The core combinatorial problem is therefore

> **SL**(k): Assume the random variables X_i are drawn from some finite set Σ. Given the conditional entropy $H(X|\Pi)$ for each variable X and each set of k parents Π, find a directed acyclic graph G which minimizes cost(G) subject to the constraint that all of its indegrees are at most k.

SL is shorthand for "structure learning". In general, SL(k) might be used as a subroutine in a model selection procedure which finds good nets for different values of k and then chooses between them by appropriately penalizing large models. We will mostly be concerned with the cases $k = 1$ and $k = 2$.

2.3 Other learning problems

The machine learning literature would describe SL as addressing the *unknown structure, fully-observable* case. This means that the structure G needs to be learned and that there are no hidden variables, that is to say the samples contain values for all n attributes.

The use of hidden variables can lead to significantly more compact probabilistic nets. For instance, if n-tuples $(X_1, \ldots, X_n)$ are generated from a mixture of two product distributions over $\{0,1\}^n$ then a fully-observable probabilistic net with nodes $X_1, \ldots, X_n$ will typically require a large amount of connectivity to accurately model this data. If, however, a single hidden node Y is introduced then a very simple structure (Figure 1) will suffice. Here Y functions as a mechanism for distinguishing clusters in the data. A different and

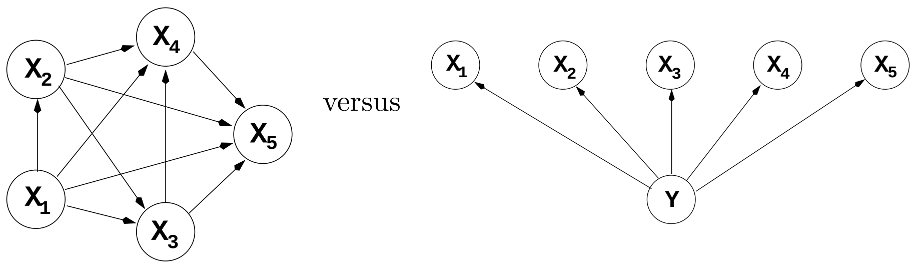


Figure 1: A single hidden node can tremendously simplify a probabilistic net.

less subtle use of hidden nodes, for storing intermediate results to reduce indegree, was mentioned in the introduction (Figure I.3).

Despite the tremendous importance of hidden variables, almost nothing is known about when and how to use them given "observable" data, except in a few of the simple clustering scenarios that we have mentioned. We too will steer clear of this difficult subject.

In the *fixed structure* scenario, the underlying graph of the probabilistic net has already been constructed by "experts" and cannot be changed even if it poorly matches the actual data. All that remains is to assign the conditional probabilities. If these take the form of CPTs, then the maximum-likelihood choice is to set them to their empirically observed values. Setting the parameters of a noisy-OR gate is only slightly more complicated, since the space of possibilities ranked by likelihood can be shown to be concave.

3 SL(k)

Once again, in the SL(k) problem we assume we know $H(X|Z_1, \ldots, Z_k)$ for all choices of $X, Z_1, \ldots, Z_k$, and the goal is to find a directed acyclic graph G which has all indegrees $\leq k$, so as to minimize

$$\text{cost}(G) = \sum_{i=1}^{n} H(X_i|\Pi_i^G).$$

We could pick the best k parents for each node X_i: the parents which convey the most information about X_i, which leave the least amount of randomness in X_i. The problem is that the resulting graph might not be acyclic. Thus the two conditions which together make the optimization task hard are the degree restriction and acyclicity. Remove either of these, and the problem is trivial.

Example 1 Suppose X_1 by itself looks like a random coin flip but its value is determined completely by X_2 and X_3. That is, $H(X_1) = 1$ but $H(X_1|X_2, X_3) = 0$. There is then a temptation to add edges from X_2 and X_3 to X_1, but it might ultimately be unwise to do so because of problematic cycles that this creates. ▮

Example 2 Let X_2, X_3, and X_4 be independent random coin flips, with $X_1 = X_2 \oplus X_3 \oplus X_4$, where $\oplus$ denotes exclusive-or. Then adding edges from X_2, X_3, and X_4 to X_1 will reduce the entropy of X_1 by one. However, any proper subset of these three parents provides no information about X_1; for instance, $H(X_1|X_2, X_3) = H(X_1) = 1$. ▮

The cost of a candidate solution can lie anywhere in the range $[0, O(n)]$. *The most interesting situations are those in which the optimal structure has very low cost (low entropy), since it is in these cases that there is structure to be discovered.*

3.1 Previous work

The central positive result in this field was the very first one, when in 1967 Edmonds demonstrated an algorithm for SL(1). Directed graphs of indegree bounded by one are called *branchings*. Edmonds showed that the maximum-likelihood branching can be found in quadratic time by a simple greedy procedure. In fact, the symmetric relationship

$$H(X) - H(X|Y) = H(Y) - H(Y|X)$$

(the amount by which knowing Y decreases the entropy of X equals the amount by which knowing X decreases the entropy of Y) makes it possible to reduce SL(1) to a single minimum spanning tree computation. This was observed by Chow and Liu (1968), with the result that branchings are often called *Chow-Liu trees* by artificial intelligence researchers.

The next candidate, SL(2), was shown to be NP-hard by Chickering (1996). His proof assumes a Bayesian setting in which there is a prior distribution on graph structures, but the techniques can be adapted to our maximum-likelihood framework.

The algorithms used to solve SL(k) in practice are generally local search heuristics, including some based on gradient descent (Russell, Binder, Koller and Kanazawa, 1995) and EM (Friedman, 1997). These procedures start with some initial guess of a structure, and then incrementally change it by adding and removing single edges. They will also accommodate hidden nodes. Example 2 illustrates one inherent limitation of such schemes.

3.2 SL(2) is hard to approximately solve

The NP-hardness of SL(2) closes one door but opens many others. Over the last three decades, provably good approximation algorithms have been developed for a whole host of NP-complete optimization problems. Do such algorithms exist for learning structure? We now present a further negative result showing that for some constant $c > 1$, unless $\mathrm{P} = \mathrm{NP}$ there is no polynomial-time procedure which *c-approximates* SL(2), that is, which consistently returns a graph whose cost is within a multiplicative factor c of optimal. From a practical viewpoint this is a weak result, since it does not rule out the possibility of, say, 2-approximation, or $O(\log n)$-approximation, either of which would be a tremendous boon. Its importance lies instead in bringing into sharper focus some of the aspects of SL(2) that make it a difficult problem, and thereby providing valuable insight into the hurdles that must be overcome by good algorithms.

Theorem 1 *There exists some constant $c > 1$ for which c-approximating SL(2) is an NP-hard optimization problem.*

Proof. The proof takes the form of a triplet of reductions,

$$\mathrm{SAT} \longrightarrow \mathrm{VC}(4) \longrightarrow \mathrm{FVS}' \longrightarrow \mathrm{SL}(2).$$

These problems are defined in Figure 2. The first reduction is standard (Arora and Lund, 1996; Papadimitriou and Yannakakis, 1991; Alimonti and Kann, 1997). Specifically, it is well-known that there is a universal constant $\epsilon > 0$, an efficiently computable function τ which sends instances of SAT to instances of VC(3), and an efficiently computable integer-valued function f on instances of SAT, such that

$$\begin{aligned} \phi \in \mathrm{SAT} \quad &\Longrightarrow \quad \mathrm{VC}(3)(\tau(\phi)) = f(\phi), \quad \text{and} \\ \phi \notin \mathrm{SAT} \quad &\Longrightarrow \quad \mathrm{VC}(3)(\tau(\phi)) > f(\phi)(1+\epsilon). \end{aligned}$$

This is often described by saying that VC(3) has an *approximability gap* of $1+\epsilon$. A similar result must hold for VC(4) since the graphs produced by τ can be altered to be four-regular by repeatedly connecting the gadget in Figure 3 to pairs of distinct vertices u, v. The gadget itself has a vertex cover of size four, regardless of the status of the two vertices it is hooked into. We will use the fact that $f(\phi)$ is always at least some constant fraction of the number of vertices in $\tau(\phi)$; call this constant $r \in (0, 1)$.

The second reduction converts instances $G = (V, E)$ of VC(4) to instances $G' = (V', E')$ of FVS$'$. This is managed by first replacing each undirected edge $\{u, v\}$ in E by a

SAT: Boolean satisfiability

Input: A Boolean formula ϕ in conjunctive normal form.

Question: Is there some assignment to the variables which satisfies ϕ?

VC(k): Vertex cover

Input: An undirected graph (V, E) whose nodes all have degree exactly k.

Output: A set of vertices $S \subset V$ such that every edge in E has at least one endpoint in S.

Objective: Minimize $|S|$.

FVS$'$: Restricted feedback vertex set

Input: A directed graph $G = (V, E)$ in which each vertex has indegree exactly two and outdegree either one or four. It is also guaranteed that G contains no triangles, that is, no undirected cycles of length three.

Output: A set of vertices $S \subset V$ whose removal makes G acyclic.

Objective: Minimize $|S|$.

Figure 2: SAT and various NP optimization problems.

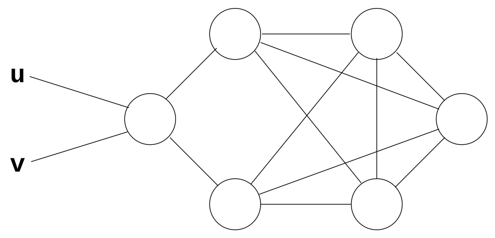


Figure 3: This gadget increases the degree of u and v by one.

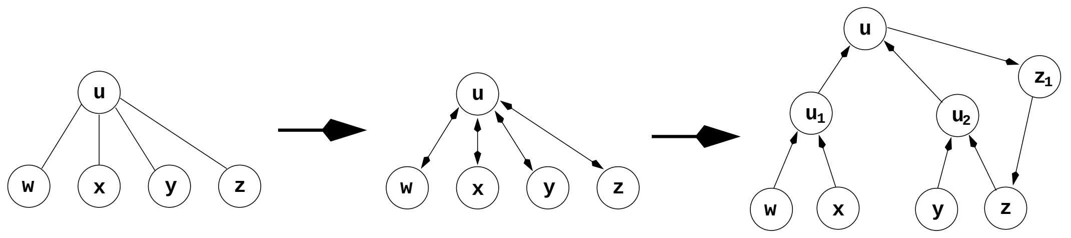


Figure 4: Mapping VC(4) to FVS$'$.

directed cycle $u \longleftrightarrow v$ in E', and then reducing the indegree of each node u from four to two by introducing two intermediate nodes u_1 and u_2 which lead to u and each of which handles two of u's inputs (Figure 4). The number of nodes is tripled, so the optimal solution to G' has size at least $\frac{1}{3}r|V'|$, while the approximability gap remains $1+\epsilon$.

We can finally start on the main reduction. We have chosen FVS$'$ as a starting point because, like SL(2), it requires finding an acyclic subgraph. Given an n-node graph $G=(V,E)$ as input to FVS$'$, we will create a very similar directed graph G' which specifies a probability distribution P. This distribution (more specifically, conditional entropy values generated from it) will be used as input to SL(2).

We have already seen how directed acyclic graphs can generate probability distributions. However, the graph G' will contain many cycles. We will ensure that its induced distribution P is well-defined by making certain that influences do not propagate very far in G'. That is, all correlations are local, or equivalently, any node (random variable) in G' can only affect the values of nodes very close to it, so there is no fear of influences travelling around cycles of G'.

An algorithm for SL(2) will attempt to model P by some directed acyclic graph G'' whose indegrees are all at most two. We will show that G'' can be used to produce a good solution for the original FVS$'$ instance. In particular, the parents of node $u \in G''$ will be either the parents of u in G (in which case the FVS$'$ solution retains node u), or a special set of default parents (in which case the FVS$'$ solution discards node u). These defaults are extra random variables whose behavior is coordinated through the gadget shown in Figure 5.

In this device, A_1 and A_2 are random $B(\frac{1}{2})$ coin flips ($B(p)$ stands for Bernoulli(p), that is, a $\{0,1\}$-valued random variable with probability p of being one), $C_1 = (A_1 \wedge A_2) \oplus B(q)$, $C_2 = (A_1 \vee A_2) \oplus B(q)$, and $L = C_1 \oplus C_2$, for some specific $q \in (0, \frac{1}{2})$ whose value we

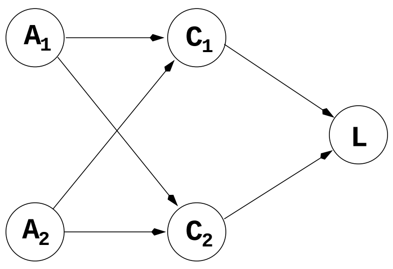


Figure 5: Gadget used in the reduction from FVS$'$ to SL(2).

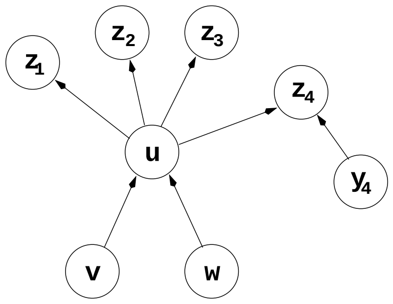
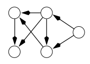
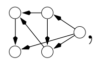


Figure 6: A node u in G.

shall not dwell upon. The entropy of the gadget's distribution is $H_0 = 2H(\frac{1}{2}) + 2H(q) = 2 + 2H(q)$. Any attempt to model the same distribution by a different degree-two-bounded structure in which L has less than two inedges will increase the cost to $H_0 + \Delta$, where $\Delta = \min\{\frac{1}{2} - \frac{1}{2}H(q), H(2q(1-q)) - H(q)\} > 0$ is some small constant. Moreover, this suboptimal cost can be achieved by an arrangement,  or , in which there are no edges into L.

We can now describe the directed graph G' in detail. Pick any node $u \in G$, with inedges from v and w and outedges to $z_1, \ldots, z_l, l \leq 4$ (Figure 6). G' will contain a gadget for u, but the L node of this gadget, denoted L_u, will be incorporated into a special random variable X_u. The remaining gadget nodes, $A_{1u}, A_{2u}, C_{1u}, C_{2u}$, will exist as Boolean random variables. In other words, there are five nodes corresponding to u in G': X_u and the four gadget nodes (Figure 7).

X_u has 64 possible values which can be segregated into six binary-valued fields, L_u, I_u, and $O_u^{(1)}, \ldots, O_u^{(4)}$. The O_u's are output fields containing information to be passed on to the X_{z_i}. Each $O_u^{(i)}$ has distribution $B(p)$, for some $p \in (0, \frac{1}{2})$, independent of everything except I_{z_i}, to which it is passed. If u has less than four outedges, then some of these output

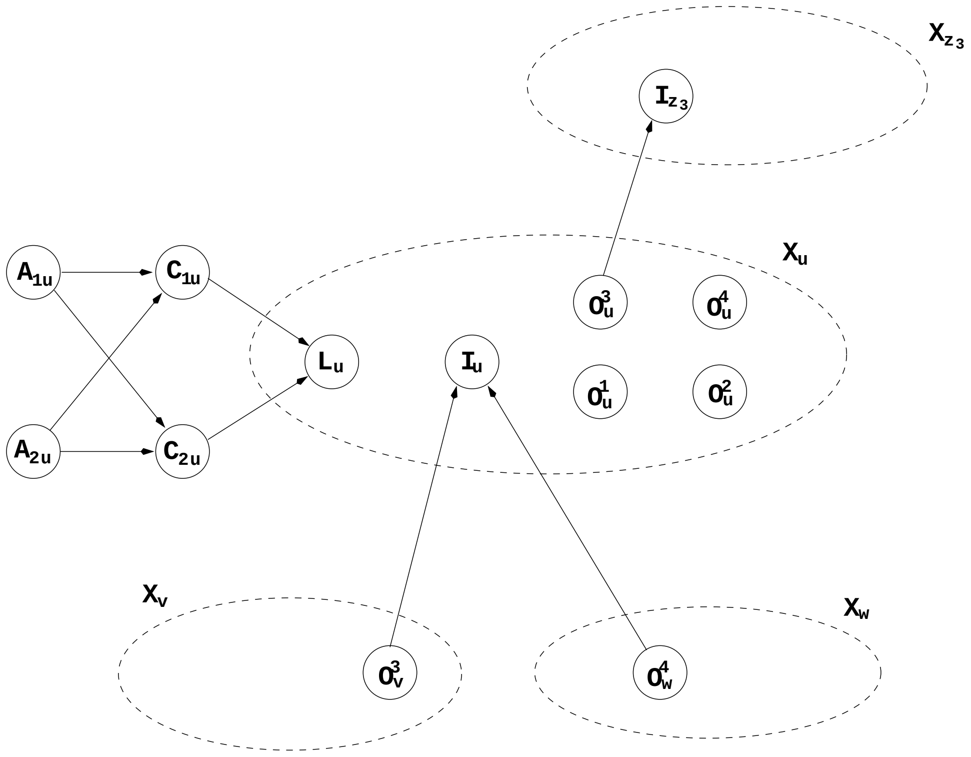


Figure 7: What u becomes in G'.

fields contain random values that never get sent anywhere. I_u receives inputs from the output fields of X_v and X_w, and its value is their exclusive-or. X_v and X_w provide a lot of information about X_u. The next most informative set of parents for X_u is $\{C_{1u}, C_{2u}\}$.

G' contains many cycles, of course, but specifies a unique probability distribution P. To see this, notice that no piece of information can travel very far in this graph – it originates in an O node and moves to an I node, where it is forgotten. To sample from this distribution, first generate values for all the gadgets, then for all the O nodes, and finally set each I node based upon the O nodes of its parents.

We now ask an SL(2) algorithm to find the best indegree-two network that models P. How will this algorithm pick the parents of each vertex? By construction, the only pairs of nodes which give any information at all about X_u must be chosen from the nodes of u's gadget, and the nodes X_w, X_v, X_{z_i}, X_{y_i}, and $X_{y'_i}$, where y_i is the other parent of z_i in G, and y'_i is a child of y_i in G. By considering the various combinations in turn, we find that there are effectively just two sensible choices for X_u's parents:

- $\{X_v, X_w\}$ (v and w being the parents of u in the original graph), which costs (a total conditional entropy of) some $H_1 = H_0 + \Delta + 4H(p)$ for the five nodes affiliated with u and can potentially produce cycles; and
- $\{C_{1u}, C_{2u}\}$, which entails an entropy cost of $H_2 = H_0 + 4H(p) + H(2p(1-p))$, and can never result in the formation of cycles.

Any other choice of parents incurs cost at least $H_3 = H_0 + \Delta + 4H(p) + \min\{H(2p(1-p)) - H(p), 2H(p) - H(2p(1-p))\}$. By setting p and q carefully, for instance $p = 0.07, q = 0.25$, we ensure that $H_3 > H_2 = H_1 + \delta$, for a small constant $\delta > 0$.

Thus an SL(2) algorithm will always be able to choose two parents for each node X_u. As far as possible it ought to choose the node's parents in the original graph G. This corresponds to retaining that node for FVS$'$ purposes. Otherwise, the parents might as well be chosen from that node's gadget, which corresponds to removing the node in G. More precisely, a solution of FVS$'$(G) in which j nodes are removed corresponds to a solution $G'' \subset G'$ of the SL(2) problem with $\text{cost}(G'') = nH_1 + j\delta$. Since δ and H_1 are constants, and since instances of FVS$'$ with optimal solutions of size at least $\frac{1}{3}rn$ are hard to approximate within a constant, we see that SL(2) is also hard to approximate within a constant. ▮

The next question is whether there exists an even more damaging hardness result. Consider the following generalization of SL.

> **HT**: Given a non-negative weight $w(X, \Pi)$ for each variable X and each set of k parents Π, find a directed acyclic graph G which minimizes $\sum_i w(X_i, \Pi_i^G)$ subject to the constraint that all of its indegrees are at most k.

This problem is hard to approximate within $O(\log n)$, by reduction from Set Cover (Höffgen, 1993). Therefore, any algorithm which seeks a better performance guarantee for SL must rely upon the fact that the weights $w(X, \Pi)$ are not arbitrary but are conditional entropies. The proofs of the next chapter will make heavy use of this.

Open Problem 1 Standard tricks for reducing fan-in and fan-out can be deployed to show that FVS$'$ is not substantially easier to approximate than the general FVS problem, for which the best known algorithm (Seymour, 1995) achieves an approximation ratio of $O(\log n \log \log n)$. However, our proof technique can demonstrate at most an $O(1)$ approximability gap for SL because the distribution P created by the reduction has entropy $\Omega(n)$. Reducing the entropy of P would require allowing its dependency graph to contain long-range (that is, non-local) influences. Can these be introduced in such a way that there are no inconsistencies around cycles?

Conversely, is there a constant-factor approximation algorithm for SL? For instance, the gadget in Figure 5 has the interesting property that any optimal arrangement of the edges which satisfies the degree bound must include the edges $C_1 \rightarrow L$ and $C_2 \rightarrow L$. Is there some generic way to identify these connections as being particularly valuable? Another direction of attack might be to initially limit attention to distributions in which all correlations are deterministic, that is, any set of parents Π either completely determines X or gives no information whatsoever about X. An example follows.

Example 3 A useful test distribution for greedy heuristics is one which is produced by a directed graph (Figure 8) with $2\sqrt{n}$ sources $X_1, \ldots, X_{\sqrt{n}}, Y_1, \ldots, Y_{\sqrt{n}}$, each i.i.d. $B(\frac{1}{2})$. The sources generate $X_{ij} = X_i \oplus X_j$ and $Y_{ij} = Y_i \oplus Y_j$, for all i, j such that $|i - j|$ is odd. Finally all combinations $Z_{ij} = X_{ij} \oplus Y_{ij}$ are present. There are a total of about $2\sqrt{n} + \frac{3}{4}n$ random variables.

The optimal structure shown in the figure has entropy $2\sqrt{n}$. There are no correlated triples of variables other than those of the types mentioned. For instance, there are no

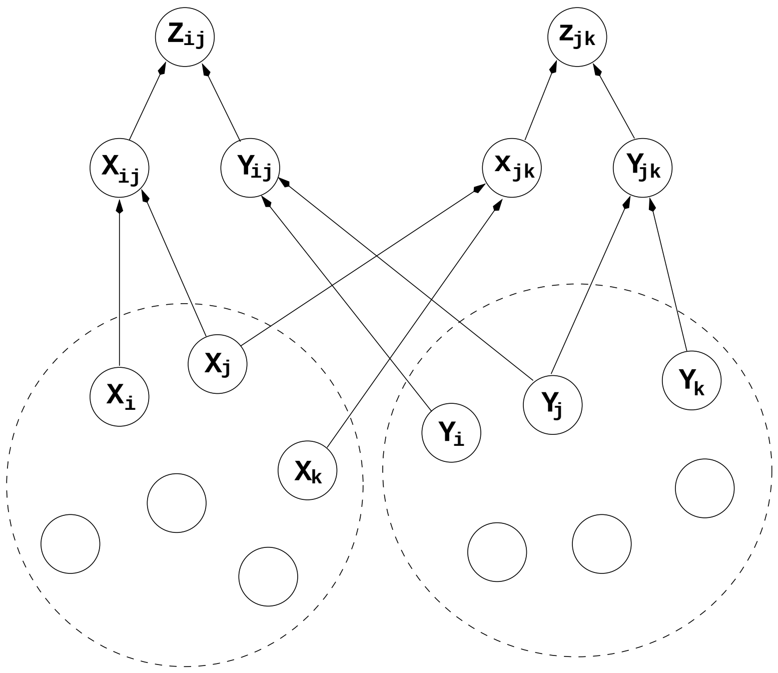


Figure 8: Finding the $2\sqrt{n}$ source variables might not be easy.

relationships $X_{ij} \oplus X_{ik} = X_{jk}$ because at least one of the differences $|i-j|, |i-k|, |j-k|$ must be even. Thus any greedy scheme which iteratively identifies strong correlations and then chooses arbitrarily amongst them will end up picking $\theta(n)$ of the X_{ij} or Y_{ij} as sources, giving an overall entropy of $\theta(n)$. ▮

Open Problem 2 In this deterministic setting, SL reduces to finding sources of the optimal structure. To what extent does this "source discovery" problem capture the hardness of SL?

Chapter V

Learning polytrees

1 Why polytrees?

The theoretical terrain of the structure learning problem so far contains just two major landmarks: it is easy to learn branchings and difficult to learn general probabilistic nets of bounded indegree. With these two endposts in mind, it is natural to ask whether there is some subclass of nets which is more general than branchings, and for which a provably good learning algorithm can be found. Even if this learning problem is NP-hard, it may well be the case that a good approximation algorithm exists. We will now take a small step in this direction by considering the task of learning polytrees from data. Polytrees have more expressive power than branchings, and have turned out to be an important class of directed probabilistic nets, largely because they permit fast exact inference (Pearl, 1988). In strict analogy with SL, we define

> **PT**: Assume the random variables X_i are drawn from a finite set Σ. Given the conditional entropy $H(X|\Pi)$ for each variable X and each set of parents Π, find a polytree G which minimizes $\text{cost}(G)$.

In deference to sample complexity considerations, we will also consider

> **PT**(k): just like PT, except that each node is allowed at most k parents (we shall call these *k-polytrees*).

It is not at all clear that PT is any easier than SL. However, we will see that, first, the optimal branching constitutes a provably good approximation to the best polytree, and second, it is not possible to do very much better, because there is some constant $c > 1$

for which it is NP-hard to c-approximate PT(2). That is to say, if on input I the cost of the optimal solution to PT(2) is denoted $OPT(I)$, then it is NP-hard to consistently find 2-polytrees of cost $\leq c \cdot OPT(I)$.

There has been some work on reconstructing a polytree given data which actually comes from a polytree distribution – for instance, by Geiger, Paz and Pearl (1990) and by Acid, de Campos, González, Molina and Pérez de la Blanca (1991). We, on the other hand, are trying to approximate an *arbitrary* distribution by a polytree.

2 An approximation algorithm

We will show that the optimal branching has cost (and therefore log-likelihood) close to that of the optimal polytree. They are different by at most a multiplicative factor which is not an universal constant but depends upon the nature of the individual attributes (nodes). For instance, if the attributes individually resemble fair coin flips, then the cost of the optimal branching is always at most twice that of the optimal polytree. This is a very significant performance guarantee, given that this latter cost can be as low as one or as high as n, and instantly gives us an $O(n^2)$-time approximation algorithm for learning polytrees.

2.1 A simple bound

The following definition will be crucial for the rest of the development.

Definition For any probability distribution over variables $X_1, \ldots, X_n$, let $U = \max_i H(X_i)$ (that is, the highest entropy of an individual node) and $L = \min_i H(X_i)$.

Example 1 If all the variables are Boolean and each by itself is a random coin flip, then $U = L = 1$. Similarly, if some variables have values in the range $\{1, 2, \ldots, 100\}$ and the rest are Boolean, and each variable by itself looks pretty uniform-random (within its range), then $U \approx \log_2 100 < 7$ and $L \approx 1$. It is always true that the cost of the optimal polytree (or the optimal branching, or the optimal directed probabilistic net) lies in the range $[L, nU]$. ▮

We warm up with an easy theorem which provides useful intuition about PT.

Theorem 1 *The cost (negative log-likelihood) of the optimal branching is at most* $(1 + \frac{U}{L})$ *times than that of the optimal polytree.*

Proof. Let the optimal polytree have total cost H^* (we have chosen the letter H because the cost is essentially an entropy rating, and it is helpful to think of it as such). We will use this polytree to construct a branching of cost $\leq H^*(1+\frac{U}{L})$, and this will prove the theorem.

In the optimal solution, let S denote the vertices with indegree more than one. Since the total number of edges in the structure is at most $n-1$, *each node of high indegree is effectively stealing inedges away from other nodes* (cf. Example IV.1). In particular therefore, the polytree must have at least $|S|+1$ sources (that is, nodes with no parents), implying that $H^* \geq |S|L$.

If we remove the edges into the vertices of S (so that they become sources in the graph), we are left with a branching. Each vertex of S has entropy at most U, and so the cost of this branching is $\leq H^* + |S|U \leq H^*(1+\frac{U}{L})$. ▮

Thus in cases where the nodes have approximately equal individual entropies, the optimal branching is at most about a factor of two worse than the optimal polytree, in terms of log-likelihood. What happens when the ratio $\frac{U}{L}$ is pretty large? This is especially a danger when different variables have vastly different ranges. Over the course of the next few lemmas, we will improve the dependence on $\frac{U}{L}$ to $O(\log \frac{U}{L})$, and we will show that this is tight. We start with a very crude bound which will be useful to us later.

2.2 A better performance guarantee

First we establish some simple

Notation. Let the optimal polytree have cost H^*, as before. *Whenever we talk about parents and ancestors henceforth, it will be with respect to this solution.* We say X is a *parent* of Y if there is an edge from X to Y; the definitions of *child* and *ancestor* follow from this. A *source* is a node with no inedges and a *sink* is a node with no outedges. Denote by T_X the induced subgraph consisting of node X and its ancestors. That is, T_X is a (directed) subgraph of the polytree such that its only sink is X and all the other nodes in it have directed paths to X (Figure 1). Let $|T_X|$ then signify the number of nodes in this subtree. For each node Z with parents Π, let $\Delta(Z) = H(Z|\Pi)$, the entropy that remains in Z even after its parents are known. Clearly $H^* = \sum_Z \Delta(Z)$. Extend Δ to subgraphs T_Z in the natural way: $\Delta(T_Z) = \sum_{X \in T_Z} \Delta(X)$.

The ensuing discussion will perhaps most easily be understood if the reader thinks of $\Delta(Z)$ as some positive real-valued attribute of node Z (such that the sum of these values

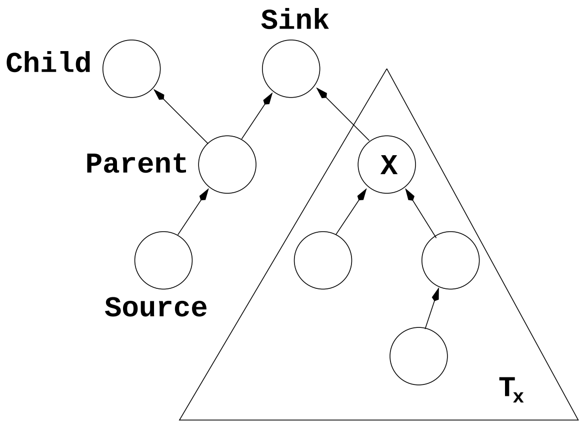


Figure 1: Notation

over all nodes is H^*) and ignores its original definition in terms of Z's parents in the optimal polytree. This is because we will be doctoring the optimal polytree while leaving the $\Delta(Z)$ fixed. We start with a few examples to clarify the notation.

Example 2 Suppose that in the optimal polytree, node X has parents $Y_1, \ldots, Y_k$. Then the subtrees $T_{Y_1}, \ldots, T_{Y_k}$ are disjoint parts of T_X and therefore

$$\sum_{i=1}^{k} \Delta(T_{Y_i}) \quad \leq \quad \Delta(T_X) \quad \leq \quad H^*.$$

Decompositions of this kind constitute the bread and butter of the proofs that follow. ▮

Example 3 For any node X, we can upper-bound its individual entropy,

$$H(X) \quad \leq \quad \Delta(T_X),$$

since the right-hand term is the total entropy of the subtree T_X which includes X. Here is another way to think of it: denote by S the variables (including X) in the subtree T_X. Then T_X is a polytree over the variables S, and so $\Delta(T_X)$ is at least the inherent entropy $H(D|_S)$ of the target distribution D restricted to variables S, which in turn is at least $H(X)$. ▮

Given the optimal polytree, we need to somehow construct a branching from it which approximates it well. Nodes with zero or one parent can remain unchanged, but for each node Z with parents $Y_1, Y_2, \ldots, Y_l, l \geq 2$, we must eliminate all but one inedge. Naturally, *the parent we choose to retain should be the one which contributes the most information to*

Z. The information lost by removing parent Y_i is at most its entropy $H(Y_i)$, which in turn is upper-bounded by $\Delta(T_{Y_i})$ (see Example 3). Therefore, we will use the simple rule: *retain the parent with highest* $\Delta(T_{Y_i})$. Formally, the increase in cost occasioned by removing all parents of Z except Y_j can be bounded by

$$H(Z|Y_j) - H(Z|Y_1, Y_2, \ldots, Y_l) \;\leq\; \sum_{i \neq j} H(Y_i) \;\leq\; \sum_{i \neq j} \Delta(T_{Y_i}) \;=\; \sum_i \Delta(T_{Y_i}) - \max_i \Delta(T_{Y_i}).$$

This motivates the following

Definition For any node Z with $l \geq 2$ parents $Y_1, \ldots, Y_l$ in the optimal polytree, let the charge $C(Z)$ be $\sum_i \Delta(T_{Y_i}) - \max_i \Delta(T_{Y_i})$. For nodes with less than two parents, $C(Z) = 0$.

So we apply our rule to each node with more than one parent, and thereby fashion a branching out of the polytree that we started with. All these edge removals are going to drive up the cost of the final structure, but by how much? Well, there is no increase for nodes with less than two parents. For other nodes Z, we have seen that the increase is at most $C(Z)$.

In this section, we will charge each node Z exactly the amount $C(Z)$, and it will turn out that the total charges are at most $\frac{1}{2}H^* \log n$. Most of the work is done in the following

Lemma 2 *For any node Z, the total charge for all nodes in subgraph T_Z is at most $\frac{1}{2}\Delta(T_Z) \log |T_Z|$.*

Proof. Let $C(T_Z) = \sum_{X \in T_Z} C(X)$ denote the total charge for nodes in subgraph T_Z. We will use induction on $|T_Z|$. If $|T_Z| = 1$, Z is a source and trivially $C(T_Z) = 0$. So, assume the claim is true for all subtrees of size less than $|T_Z|$. If Z has just one parent, say Y, then $C(T_Z) = C(T_Y)$, and we are done. Assume, then, that Z has parents $Y_1, \ldots, Y_l, l \geq 2$, and let $\delta_i = |T_{Y_i}|/|T_Z|$. Then $\delta_1 + \cdots + \delta_l \leq 1$, and

$$\begin{aligned}
C(T_Z) &= \sum_i C(T_{Y_i}) + C(Z) \\
&\leq \sum_i \tfrac{1}{2}\Delta(T_{Y_i}) \log \delta_i |T_Z| + \sum_i \Delta(T_{Y_i}) - \max_i \Delta(T_{Y_i}) \\
&\leq \tfrac{1}{2}\Delta(T_Z) \log |T_Z| + \sum_i \Delta(T_{Y_i})(1 + \tfrac{1}{2} \log \delta_i) - \max_i \Delta(T_{Y_i}) \\
&\leq \tfrac{1}{2}\Delta(T_Z) \log |T_Z|,
\end{aligned}$$

where the second line applies the inductive hypothesis to subtrees $T_{Y_1}, \ldots, T_{Y_l}$, the third line follows from $\Delta(T_Z) \geq \sum_i \Delta(T_{Y_i})$ (because these l subtrees are disjoint), and the fourth line is the result of (1) ignoring all δ_i smaller than $\frac{1}{4}$, (2) applying Jensen's inequality, and (3) using the fact that $\max_i a_i$ dominates any convex combination of $\{a_i\}$. ▮

Now we apply this to selected subgraphs of the optimal polytree. Our aim, again, is to bound the additional cost incurred by removing edges necessary to convert the optimal polytree into a branching. Specifically, we want to show

$$\sum_{i=1}^{n} C(X_i) \quad \leq \quad \tfrac{1}{2} H^* \log n \quad = \quad \sum_{i=1}^{n} \Delta(X_i) \cdot \tfrac{1}{2} \log n.$$

Theorem 3 *The cost of the optimal branching is at most* $(1 + \frac{1}{2}\log n)$ *times the cost of the optimal polytree.*

Proof. Suppose the optimal polytree were the union of a few *disjoint* subtrees $T_{Z_1}, \ldots, T_{Z_l}$ (equivalently, suppose each connected component had just one sink). Then we would be in business, because we could apply the previous lemma to each of these subtrees and sum the results. We would get

$$\begin{aligned} \sum_{i=1}^{n} C(X_i) &= \sum_{i=1}^{l} C(T_{Z_i}) \\ &\leq \sum_{i=1}^{l} \tfrac{1}{2} \Delta(T_{Z_i}) \log |T_{Z_i}| \\ &\leq \sum_{i=1}^{n} \tfrac{1}{2} \Delta(X_i) \log n \\ &= \tfrac{1}{2} H^* \log n, \end{aligned}$$

where the second line uses Lemma 2 and the third line uses $|T_{Z_i}| \leq n$.

If the optimal polytree is not of this convenient form, then it must have some connected component which contains two sinks X and Y such that T_X and T_Y share a common subgraph T_U (Figure 2).

Say that T_U is connected to the rest of T_Y through the edge $(U, V), V \in T_Y - T_U$. Remove this edge, and instead add the edge (X, V). This has the effect of moving $T_Y - T_U$ all the way up T_X. After this change, X is no longer a sink, and for every node Z in the graph, $\Delta(T_Z)$ (and thus $C(Z)$) has either stayed the same or increased, because each node has all the ancestors it had before and maybe a few more (recall that we are thinking of each $\Delta(W)$ as a fixed value associated with node W and that $\Delta(T_Z)$ is the sum of these values over $W \in T_Z$). It is also the case, and this will be useful later, that the indegree of

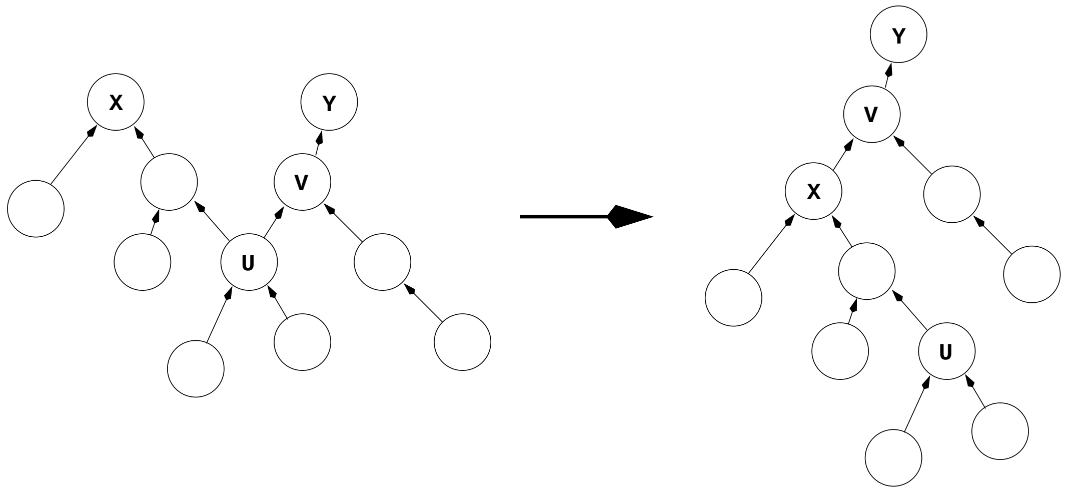


Figure 2: T_X and T_Y contain a common subtree.

each node remains unchanged. In this manner we alter the structure so that it is a disjoint union of single-sink components, and then apply the previous argument. ▮

In the optimal polytree, let n_0 denote the number of nodes with no parents (that is, the number of sources), and $n_{\geq 2}$ the number of nodes with at least 2 parents. By examining what happens to nodes of different indegree in the above proof, we notice that the theorem remains true if we replace n by $n_0 + n_{\geq 2}$, that is, if we ignore nodes with exactly one inedge.

2.3 The final improvement

In the argument so far, we have bounded the entropy of a node X by $H(X) \leq \Delta(T_X)$. This is a fairly tight bound in general, except when these entropy values start getting close to the maximum entropy U. We factor this in by using a slightly more refined upper bound.

Definition For any node Z, let $C'(Z) = \min\{C(Z), U\}$. Then $C'(Z)$ is an upper bound on the extra cost incurred at node Z due to the removal of all but one parent.

This gives us the final improvement. Once again, we start by considering a single subtree.

Lemma 4 *For any node Z, the total cost incurred by nodes in the subtree T_Z is at most $C'(T_Z) \leq (\frac{5}{2} + \frac{1}{2}\log\frac{U}{L})\Delta(T_Z)$.*

Proof. Let $T \subseteq T_Z$ denote the polytree induced by all nodes $X \in T_Z$ such that $\Delta(T_X) > U$; that is, T consists of these nodes, and any edges between them in T_Z. Note that T must be

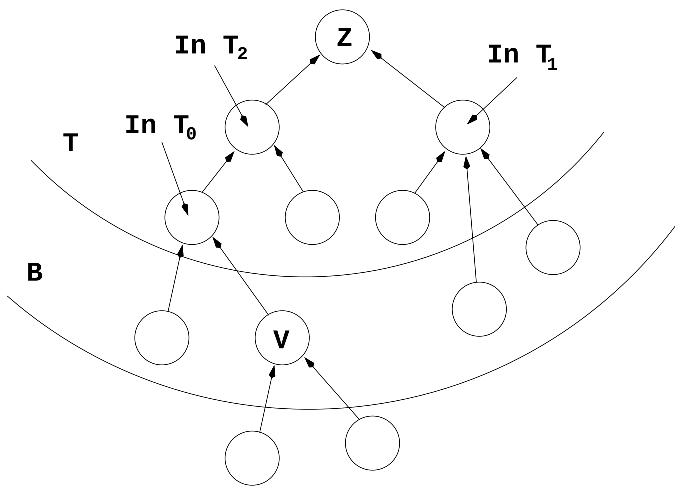


Figure 3: The regions $T, B \subset T_Z$.

connected, because if $X \in T$, and $Y \in T_Z$ is a child of X, then Y must also be in T (since $\Delta(T_Y) \geq \Delta(T_X)$). Let $T_j \subseteq T$ be those nodes of T which have j parents *in* T. And let B be the border nodes right outside T, that is, nodes which are in $T_Z - T$ and have children in T (Figure 3).

What are the charges for the various nodes in T?

(1) Each node $X \in T_1$ has all but one parent outside T, and therefore gets charged at most the sum of $\Delta(T_Y)$ over its parents Y outside T (and in B). The total charge for nodes in T_0 and T_1 is

$$\sum_{X \in T_0} C'(X) + \sum_{X \in T_1} C'(X) \quad \leq \quad \sum_{Y \in B} \Delta(T_Y) \quad \leq \quad \Delta(T_Z),$$

where the final inequality follows by noticing that the preceding summation is over disjoint subtrees of T_Z (cf. Example 2).

(2) Nodes $X \in T_2, T_3, \ldots$ have at least two parents in T, and so get charged exactly U; however, since T is a tree we know that $|T_0| \geq 1 + |T_2| + |T_3| + \cdots$, and thus

$$\sum_{X \in T_i, i \geq 2} C'(X) \quad \leq \quad U \cdot |T_0| \quad \leq \quad \sum_{X \in T_0} \Delta(T_X) \quad \leq \quad \Delta(T_Z),$$

where once again the disjoint subtrees property is used.

Thus nodes in T have total charge at most $2\Delta(T_Z)$. It remains to assess the charges for nodes in T_Z but outside T. Split these nodes into their disjoint sub-polytrees $\{T_V : V \in B\}$, and consider one such T_V. Since $\Delta(T_V) \leq U$ and each source in T_V has entropy at least L, we can conclude that T_V has at most $\frac{U}{L}$ sources and therefore at most $\frac{2U}{L}$ nodes of indegree $\neq 1$. The charge $C'(T_V)$ is therefore, by Lemma 2, at most $\frac{1}{2}\Delta(T_V)\log\frac{2U}{L}$. We sum this over the various disjoint $\{T_V : V \in B\}$ and find that the total charge for $T_Z - T$ is at most $\frac{1}{2}\Delta(T_Z)\log\frac{2U}{L}$, whereupon $C'(T_Z) \leq (\frac{5}{2} + \frac{1}{2}\log\frac{U}{L})\Delta(T_Z)$, as promised. ▮

Theorem 5 *The cost of the optimal branching is at most* $(\frac{7}{2} + \frac{1}{2}\log\frac{U}{L})$ *times the cost of the optimal polytree.*

Proof. It can be verified as in Theorem 3 that for charging purposes we may assume the optimal solution is a disjoint union of single-sink subgraphs. Apply the previous lemma to each of these subtrees in turn, and sum the results. ▮

Example 4 The bound on the ratio between the best branching and the best polytree has to depend upon $\log\frac{U}{L}$. To see this, consider a polytree which is structured as a complete binary tree (with $\frac{1}{2}n + 1$ leaves) and with edges pointing towards the single sink. All nodes are binary-valued, each internal node is the exclusive-or of its two parents, and each source has some small entropy $\epsilon = \theta(1/n)$. The nodes on the next level up each have approximately double this entropy, and so on, so that the sink has entropy about $\theta(1)$. The optimal polytree has cost $(\frac{1}{2}n + 1)\epsilon = \theta(1)$ whereas the best branching loses $\theta(\epsilon n)$ entropy on each level and therefore has cost about $\theta(\epsilon n \log n) = \theta(\log n) = \theta(\log\frac{U}{L})$. ▮

3 A hardness result

Needless to say, it is NP-hard to learn the optimal degree-bounded polytree. However, the news is considerably worse than this – we will prove a hardness barrier which rules out the possibility of a polynomial time approximation scheme. Edmonds' algorithm solves PT(1) optimally; it will now turn out that if nodes are allowed a second parent then the problem becomes hard to approximately solve.

Theorem 6 *There is some constant $c > 1$ for which the problem of finding a 2-polytree whose cost is within c times optimal is an NP-hard optimization problem.*

Proof. We shall reduce from the following canonical problem ($\epsilon > 0$ is some universal constant).

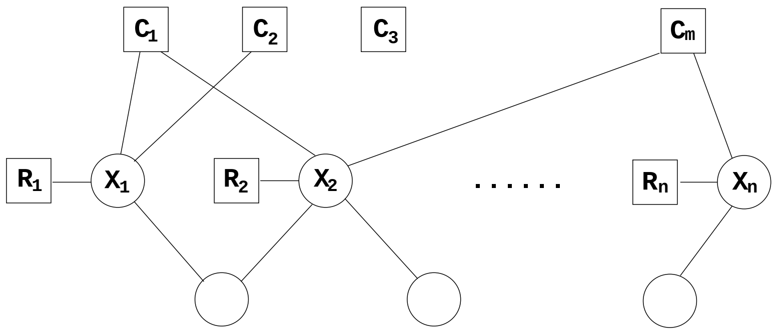


Figure 4: Broad view of the distribution created from ϕ.

MAX3SAT(3)

Input: A Boolean formula ϕ in conjunctive normal form, in which each clause contains at most three literals and each variable appears exactly three times. It is guaranteed that either there is an assignment which satisfies all the clauses (that is, ϕ is satisfiable) or no assignment satisfies more than a $(1 - \epsilon)$ fraction of the clauses.

Question: Is ϕ satisfiable?

This is a decision (true/false) version of the corresponding approximation problem (find an assignment which comes close to maximizing the number of clauses satisfied). In a celebrated result of the early 1990s, it was shown that this problem is NP-hard; a good starting point for further details is a survey article by Arora and Lund (1996).

Suppose we are given a formula $\phi = \mathcal{C}_1 \wedge \mathcal{C}_2 \wedge \cdots \wedge \mathcal{C}_m$ over variables $x_1, \ldots, x_n$, and with the stated guarantees. We will show that there is some constant $c > 1$, depending only upon ϵ, such that any algorithm which can learn a 2-polytree within a multiplicative factor c of optimal can also be used to decide whether ϕ is satisfiable.

For this reduction we construct a probability distribution out of the formula ϕ. The distribution will be specified by the probabilistic net shown in Figure 4. The edges in this graph represent correlations. Circles denote regular nodes. The squares are slightly more complicated structures whose nature will be disclosed later; for the time being they should be thought of as no different from the circles. There are three layers of nodes.

1. The top layer consists of i.i.d. $B(p)$ random variables, one per clause, for some $p \in (0, 1)$. This entire layer has entropy $mH(p)$.

2. The middle layer contains two nodes per variable x_i: a principal node X_i which is

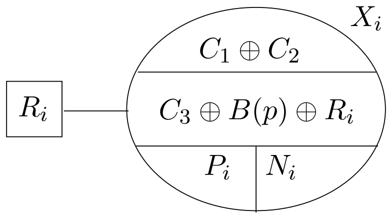


Figure 5: The R and X nodes for a specific variable in ϕ.

correlated with the nodes for the three $\mathcal{C}_j$'s in which x_i appears, and a $B(\frac{1}{2})$ satellite node called R_i. Let us focus on a particular variable x_i which appears in clauses $\mathcal{C}_1, \mathcal{C}_2$ with one polarity and $\mathcal{C}_3$ with the opposite polarity – this is the general case since each variable appears exactly three times. For instance, $\mathcal{C}_1$ and $\mathcal{C}_2$ might contain x_i while $\mathcal{C}_3$ contains $\overline{x}_i$. Then the nodes R_i, X_i will have the structure shown in Figure 5. Here P_i and N_i are both $B(\frac{1}{2})$, and stand for "previous" and "next." Each R_i is Boolean with entropy one and each X_i is a four-tuple of Boolean values, with entropy $H(2p(1-p)) + 3$. Thus the sum of the entropies of the second layer nodes is $n(H(2p(1-p)) + 4)$.

3. The third layer (of $n-1$ nodes) joins together consecutive nodes X_i, X_{i+1} of the second layer. The value of the ith node in this level, $1 \leq i \leq n-1$, is $N_i \oplus P_{i+1}$, and the entire level has entropy $n-1$.

To sample randomly from this distribution, pick the C_i's, R_i's, P_i's, and N_i's independently and at random, and then set the X_i's based upon these.

Suppose that the learning algorithm knows the exact conditional entropies for all combinations of variables. What kind of polytree can it construct? We will show that:

- If ϕ is satisfiable then there is a 2-polytree whose cost is some value H^* which can easily be computed from ϕ.
- If ϕ is not satisfiable (and so at most $m(1-\epsilon)$ clauses can be satisfied by any assignment) then there is no 2-polytree of cost less than cH^*.

Thus, any algorithm which can find a 2-polytree within factor c of optimal, can also decide whether ϕ is satisfiable.

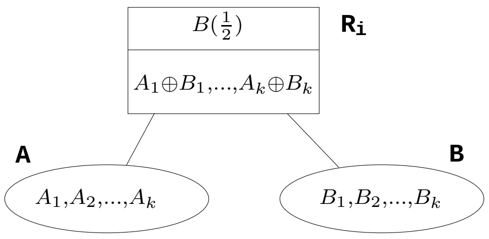


Figure 6: A device which provides two default inedges to node R_i.

How can we prove this? A good polytree must somehow tell us a satisfying assignment for ϕ. If the polytree that is learned has an edge from C_j to X_i, we will say "C_j chooses X_i" and will take this as meaning that clause $\mathcal{C}_j$ is satisfied on account of variable x_i. There are several steps towards showing this assignment is well-defined.

First we will force the square nodes to have no inedges. This is quite easy to achieve; for instance, R_1, ostensibly a single coin flip, can be rigged to avoid inedges by giving it two default inedges via the little gadget of Figure 6. Here A and B are two extra nodes, independent of everything but R_1, which each contain k i.i.d. copies of $B(q)$, called $A_1, A_2, \ldots, A_k$ and $B_1, \ldots, B_k$, for some constants $q < \frac{1}{2}$ and $k > 1$. R_1 has now expanded to include $A_1 \oplus B_1, \ldots, A_k \oplus B_k$. On account of the polytree constraint, there can be at most two edges between A, B, and R_1. The optimal choice is to pick edges from A and B to R_1. Any other configuration will increase the cost by at least $k(H(2q(1-q)) - H(q)) > 1$ (the constant k is chosen to satisfy this). These suboptimal configurations might permit inedges from outside nodes which reveal something about the $B(\frac{1}{2})$ part of R_1; however such edges can provide at most one bit of information and will therefore never be chosen (they lose more than they gain). In short, these gadgets ensure that square nodes will have no inedges from the nodes in the overall graph (Figure 4).

Next we will show that any half-decent polytree must possess all $2(n-1)$ connections between the second and third levels. Any missing connection between consecutive X_i, X_{i+1} causes the cost to increase by one and in exchange permits at most one extra edge among the X_i's and C_j's, on account of the polytree constraint. However we will see that none of these edges can decrease the cost by one.

Therefore, all the consecutive X_i's are connected via the third layer, and each clause-node C_j in the first level can choose at most one variable-node X_i in the second level (otherwise there will be a cycle).

We now need to make sure that the clauses which choose a variable do not suggest different assignments to that variable. In our example above, variable x_i appeared in $\mathcal{C}_1, \mathcal{C}_2$ with one polarity and $\mathcal{C}_3$ with the opposite polarity. Thus, for instance, we must somehow discourage edges from both C_1 and C_3 to X_i, since these edges would have contradictory interpretations (one would mean that x_i is true, the other would mean that x_i is false). The structure of the node X_i imposes a penalty on this combination.

Choose p so that $H(p) = \frac{1}{2}$, and define δ to be $1 - H(2p(1-p)) > 0$. Assume that we start with a structure that has no edges in the first and second layers (apart from the hidden edges in the square nodes). The goal of the learning algorithm is to add edges from the C_j's and R_i's to the X_i's which will bring the cost down as much as possible. Which edges is it likely to add? What are the possible parents of the X_i depicted above?

- Just R_i: the entropy of X_i is decreased by δ.
- R_i, and one of C_1, C_2, C_3: the entropy of X_i is decreased by $\frac{1}{2}$.
- C_1, C_2: the entropy decrease is $1 - \delta$.
- C_1, C_3 or C_2, C_3: the entropy decrease is $\frac{1}{2} - \delta$.

Thus C_1 and C_3 will not both choose variable X_i, and more generally, the assignment embodied in the edges from clause-nodes to variable-nodes will be well-defined. Each node X_i can earn an entropy reduction of δ by merely having an inedge from R_i. In addition, each satisfied clause will reduce entropy further by $\frac{1}{2} - \delta$. Suppose m' clauses are satisfiable (either $m' = m$ or $m' \leq m(1-\epsilon)$). Then the edges into the X_i's will decrease the cost of the second layer by $m'(\frac{1}{2} - \delta) + n\delta$.

In this way, if ϕ is satisfiable, then the optimal structure has some cost $H^* = \theta(m)$, whereas if only $(1-\epsilon)m$ of ϕ's clauses are satisfiable, then the optimal cost is $H^* + \theta(\epsilon m) \geq cH^*$ for some constant $c > 1$ which depends only upon ϵ and not upon ϕ. Thus PT(2) is hard to approximate within some fixed constant factor. ▮

Open Problem 1 This proof relies heavily upon the degree restriction as a source of hardness. It should be possible to produce similar effects using just the polytree constraint.

Open Problem 2 We have seen that the optimal branching, in which each node has at most one parent, is a good approximation to the optimal polytree, in terms of log-likelihood. Any algorithm with a better performance guarantee must be able to give nodes two or more parents when necessary. What are good heuristics for doing this?

Open Problem 3 Polytrees are currently the most natural class of directed graphs for which efficient inference algorithms are known. But in fact the junction tree algorithm works quickly for any directed graph whose moralized, triangulated, undirected version has very small cliques. Is there a larger class of directed graphs with simple characterization which permits fast inference and for which efficient, provably good learning algorithms can be constructed?

Chapter VI

Conclusions and open problems

1 Mixture models

The most important contribution of this thesis to the theory of mixture models is its suggestion that random projection be used to reduce the dimension of data sets. This has a wide range of benefits, the first of which is that arbitrarily high-dimensional data from a mixture of k well-separated distributions can be mapped into just $O(\log k)$ dimensions. Another especially helpful effect is a reduction in the eccentricity of the Gaussians.

Open Problem 1 Lemma II.4 bounds the eccentricity of a projected Gaussian in terms of its original eccentricity and the initial and reduced dimensions. Is this bound tight?

Mixtures of identical spherical Gaussians in high dimension are easy to learn. However, real clusters are unlikely to be spherical-Gaussian, if only because different attributes are typically measured in different units. Our approach has been to exploit the fact that random projection makes a variety of different distributions, including very ellipsoidal Gaussians, look more spherical-Gaussian, in the sense of the "weak assumption" of Chapter III. This beneficial effect deserves much more detailed treatment than it has received here. There is one question that is particularly beguiling.

Open Problem 2 Let X be a random vector chosen uniformly from a subset S of the hypercube $\{0,1\}^n$. If S is large, say $|S| = \frac{1}{2} \cdot 2^n$, is it the case that for most unit directions γ, the distribution of $\gamma \cdot X$ is close to normal? How close?

The particular promise of this approach makes it important to develop algorithms which are adept at learning mixtures of well-separated spherical Gaussians in relatively low dimension. We have analyzed one such algorithm which has the unfortunate requirement that the Gaussians all have about the same radius.

Open Problem 3 Is there a simple algorithm which will learn a mixture of spherical Gaussians with widely differing radii?

A related project is to obtain a better theoretical understanding of some of the local search heuristics which are currently used for learning mixture models.

Open Problem 4 Is there a suitable method of initialization under which EM affords strong finite-sample performance guarantees for learning spherical mixtures of Gaussians? Can such a result be shown under the weak Gaussian assumption?

2 Directed probabilistic nets

We have identified SL as the core combinatorial problem in the task of learning the maximum-likelihood directed probabilistic net from data. The central proof of Chapter IV, showing that SL(2) is hard to c-approximate for some constant $c > 1$, requires an unusual reduction in which a probability distribution is explicitly constructed from some instance of a graph problem. Can these techniques be extended to show stronger hardness results?

Open Problem 5 Can SL(2) be shown to be as hard as Set Cover or FVS? Conversely, is there any algorithm for SL(2) with a nontrivial performance guarantee?

Learning the maximum-likelihood polytree might not be any easier than SL. However, we have seen that the representational power of polytrees is not vastly superior to that of branchings. It would be a significant theoretical advance to develop a PT(2) algorithm whose performance guarantees did not rely solely upon this fact, an algorithm which could make principled decisions about when to assign a node more than one parent.

Open Problem 6 Does PT have a polynomial-time c-approximation algorithm, for some small constant c?

Appendix A

Large deviation bounds

1 Chernoff-Hoeffding bounds

These are reproduced from the book by Kearns and Vazirani (1991).

Lemma 1 (Chernoff-Hoeffding) *Let $X_1, \ldots, X_n$ be a sequence of i.i.d. Bernoulli trials with $\mathbf{E}X_i = p$. Let S_n denote the sum $X_1 + \cdots + X_n$. Then for any $0 \le \gamma \le 1$,*
(a) $\mathbf{P}(S_n > (p+\gamma)n) \le e^{-2n\gamma^2}$*;*
(b) $\mathbf{P}(S_n < (p-\gamma)n) \le e^{-2n\gamma^2}$*;*
(c) $\mathbf{P}(S_n > (1+\gamma)pn) \le e^{-np\gamma^2/3}$*; and*
(d) $\mathbf{P}(S_n < (1-\gamma)pn) \le e^{-np\gamma^2/2}$*.*

The restriction $\gamma \le 1$ makes these bounds useful only for modest deviations; for the $\gamma > 1$ regime the following, from Motwani and Raghavan (1995), can be applied.

Lemma 2 (Chernoff-Hoeffding) *Let $X_1, \ldots, X_n$ be a sequence of i.i.d. Bernoulli trials with $\mathbf{E}X_i = p$. Let S_n denote the sum $X_1 + \cdots + X_n$. Then for any $\Delta > 0$,*

$$\mathbf{P}(S_n > \Delta pn) \le \left(\frac{e^{\Delta - 1}}{\Delta^\Delta} \right)^{pn}.$$

In particular, when $\Delta \ge 2$ the right-hand side is at most $\exp(-\frac{1}{4}\, pn\Delta \ln \Delta)$.

2 Large deviation bounds for high-dimensional Gaussians

The next few results are straightforward and presumably well-known. Proofs are included for the sake of completeness.

Lemma 3 *Randomly draw s points $Y_1, \ldots, Y_s$ from $N(\mu, I_n)$. Then for any $\Delta \geq \frac{1}{\sqrt{s}}$,*

$$\mathbf{P}\left(\left\|\frac{Y_1 + \cdots + Y_s}{s} - \mu\right\| \geq \Delta\sqrt{n}\right) \leq \left(\frac{e^{s\Delta^2 - 1}}{s\Delta^2}\right)^{-n/2}.$$

Proof. Let $Z_i = Y_i - \mu \overset{d}{=} N(0, I_n)$. The mean $\frac{1}{s}(Z_1 + \cdots + Z_s)$ has distribution $N(0, \frac{1}{s}I_n)$, and its squared L_2 norm has moment-generating function $\phi(t) = (1 - \frac{2t}{s})^{-n/2}$ for $t \leq \frac{s}{2}$. By Markov's inequality,

$$\mathbf{P}\left(\left\|\frac{Z_1 + \cdots + Z_s}{s}\right\| \geq \Delta\sqrt{n}\right) \leq \frac{\phi(t)}{e^{t\Delta^2 n}};$$

the lemma follows by choosing $t = \frac{s}{2}(1 - \frac{1}{\Delta^2 s})$. ▮

Lemma 4 *Let $X_1, \ldots, X_n$ be i.i.d. $N(0,1)$ random variables. Pick any sequence of positive coefficients $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$, and define $\lambda = \frac{1}{n}(\lambda_1 + \cdots + \lambda_n)$ and $r = \sqrt{\lambda_n/\lambda_1}$. Notice that $\sum_i \lambda_i X_i^2$ has expectation λn. Then for any $0 < \epsilon \leq 1$,*

$$\mathbf{P}\Big(\Big|\sum_{i=1}^n \lambda_i X_i^2 - \lambda n\Big| > \epsilon\lambda n\Big) \quad \leq \quad 2\exp\left(-\frac{n\epsilon^2}{24r^2}\right).$$

Proof. Each X_i^2 has moment-generating function $(1-2t)^{-1/2}$, for $t \leq \frac{1}{2}$. Therefore, applying Markov's inequality,

$$\begin{aligned}
\mathbf{P}\Big(\sum_{i=1}^n \lambda_i X_i^2 > (1+\epsilon)\lambda n\Big) &= \mathbf{P}\Big(\exp(\textstyle\sum_i t\lambda_i X_i^2) > \exp(t(1+\epsilon)\lambda n)\Big) \\
&\leq \frac{\prod_i \mathbf{E} e^{t\lambda_i X_i^2}}{e^{t\lambda n(1+\epsilon)}} \\
&= \frac{1}{e^{t\lambda n(1+\epsilon)}}\left(\prod_i \frac{1}{1 - 2t\lambda_i}\right)^{1/2}
\end{aligned}$$

for $0 < t \leq \frac{1}{2\lambda_n}$. Choose $t = \frac{1}{2\lambda_n}\frac{1}{1+\epsilon/2}\ln(1+\epsilon/2)$ to bound this expression by $\exp(-\frac{\epsilon^2 \lambda n}{24\lambda_n}) \leq \exp(-\frac{n\epsilon^2}{24r^2})$. Similarly, for $t \geq 0$,

$$\begin{aligned}
\mathbf{P}\Big(\sum_i \lambda_i X_i^2 < (1-\epsilon)\lambda n\Big) &= \mathbf{P}\Big(\exp(-\textstyle\sum_i t\lambda_i X_i^2) > \exp(-t(1+\epsilon)\lambda n)\Big) \\
&\leq \frac{\prod_i \mathbf{E} e^{-t\lambda_i X_i^2}}{e^{-t\lambda n(1-\epsilon)}} \\
&= e^{t\lambda n(1-\epsilon)}\left(\prod_i \frac{1}{1 + 2t\lambda_i}\right)^{1/2}.
\end{aligned}$$

This is at most $\leq \exp(-\frac{n\epsilon^2}{8r^2})$ if you pick $t = \frac{1}{2\lambda_n}\frac{1}{1-\epsilon/2}\ln\frac{1}{1-\epsilon/2}$. ▮

Lemma 5 *Let $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ be any increasing sequence of positive coefficients, and define $\lambda = \frac{1}{n}(\lambda_1 + \cdots + \lambda_n)$ and $r = \sqrt{\lambda_n/\lambda_1}$. If U is a random unit vector in $\mathbb{R}^n$, then for any $0 < \epsilon \leq 1$,*

$$\mathbf{P}\Big(\Big|\sum_{i=1}^n \lambda_i U_i^2 - \lambda\Big| > \epsilon\lambda\Big) \quad \leq \quad \exp\Big(-\Omega\Big(\frac{n\epsilon^2}{r^2}\Big)\Big).$$

Proof. Let $X = (X_1, \ldots, X_n)$ be a vector of n i.i.d. $N(0,1)$ random variables. Then U has the same distribution as $\frac{1}{\|X\|}(X_1, \ldots, X_n)$, and

$$\sum_{i=1}^n \lambda_i U_i^2 \quad \stackrel{d}{=} \quad \frac{\lambda_1 X_1^2 + \cdots + \lambda_n X_n^2}{X_1^2 + \cdots + X_n^2}.$$

The numerator has expectation $n\lambda$ while the denominator has expectation n. The previous lemma provides large deviation bounds for each. ▮

3 An elementary proof of the Johnson-Lindenstrauss lemma

Lemma 6 (Johnson-Lindenstrauss, 1984) *Fix some $0 < \epsilon < 1$ and $0 < \delta < 1$. Pick any m data points $x_1, \ldots, x_m \in \mathbb{R}^n$. If these points are projected into a randomly chosen subspace of dimension $d \geq \frac{12}{\epsilon^2} \log \frac{m^2}{\delta}$ then with probability $> 1 - \delta$, the projected points $x_1^*, \ldots, x_m^*$ satisfy*

$$(1-\epsilon)\,\frac{d}{n} \quad \leq \quad \frac{\|x_i^* - x_j^*\|^2}{\|x_i - x_j\|^2} \quad \leq \quad (1+\epsilon)\,\frac{d}{n} \qquad \textit{for all } i \neq j.$$

The original proof of Johnson and Lindenstrauss was simplified by Frankl and Maehara (1988), using geometric insights and refined approximation techniques. The proof given here, joint work with Gupta (1999), uses elementary probabilistic techniques to obtain essentially the same results.

We start by estimating the length of a unit vector in $\mathbb{R}^n$ when it is projected into a randomly chosen d-dimensional subspace. This length has the same distribution as the length of a random unit vector projected into a fixed d-dimensional subspace. We take this fixed subspace to be that spanned by the first d coordinate vectors, for simplicity.

Let $X_1, \ldots, X_n$ be i.i.d. $N(0,1)$ random variables. Then the distribution of $Y = \frac{1}{\|X\|}(X_1, \ldots, X_n)$ is uniform over the surface of the n-dimensional unit sphere. Let the vector $Y^* \in \mathbb{R}^d$ be the projection of Y onto its first d coordinates, so $\mathbf{E}\|Y^*\|^2 = \frac{d}{n}$. We will show that $\|Y^*\|^2$ is tightly concentrated around this mean value.

Lemma 7 *Choose any $d < n$. Let $X_1, \ldots, X_n$ be i.i.d. $N(0,1)$ random variables and let Y^* denote the first d coordinates of $Y = \frac{1}{\|X\|}(X_1, \ldots, X_n)$.*
(a) If $\beta < 1$ then

$$\mathbf{P}(\|Y^*\|^2 \leq \beta d/n) \ \leq \ \beta^{d/2}\left(1 + \frac{(1-\beta)d}{(n-d)}\right)^{(n-d)/2} \ \leq \ \exp\left(\frac{d}{2}(1-\beta+\ln\beta)\right).$$

(b) If $\beta > 1$ then

$$\mathbf{P}(\|Y^*\|^2 \geq \beta d/n) \ \leq \ \beta^{d/2}\left(1 + \frac{(1-\beta)d}{(n-d)}\right)^{(n-d)/2} \ \leq \ \exp\left(\frac{d}{2}(1-\beta+\ln\beta)\right).$$

Proof. The proofs are similar to those for Chernoff-Hoeffding bounds. We use the fact that if $X \overset{d}{=} N(0,1)$, then X^2 has moment-generating function $(1-2t)^{-1/2}$, for $t < \frac{1}{2}$.

$$\begin{aligned}
\mathbf{P}(\|Y^*\|^2 \leq \beta d/n) &= \mathbf{P}(n(X_1^2 + \cdots + X_d^2) \leq d\beta(X_1^2 + \cdots + X_n^2)) \\
&= \mathbf{P}(d\beta(X_1^2 + \cdots + X_n^2) - n(X_1^2 + \cdots + X_d^2) \geq 0) \\
&= \mathbf{P}(\exp\left\{t(d\beta(X_1^2 + \cdots + X_n^2) - n(X_1^2 + \cdots + X_d^2))\right\} \geq 1) \\
&\leq \mathbf{E}(\exp\left\{t(d\beta(X_1^2 + \cdots + X_n^2) - n(X_1^2 + \cdots + X_d^2))\right\}) \\
&= \mathbf{E}(\exp\left\{td\beta X^2\right\})^{(n-d)}\mathbf{E}(\exp\left\{t(d\beta - n)X^2\right\})^d \\
&= (1-2td\beta)^{-(n-d)/2}(1-2t(d\beta-n))^{-d/2},
\end{aligned}$$

for $t \geq 0$, $td\beta < \frac{1}{2}$, and $t(d\beta - n) < \frac{1}{2}$. Choosing $t = t_0 = \frac{(1-\beta)}{2\beta(n-d\beta)}$ then completes the proof of (a). The proof of (b) is almost exactly the same. By the same calculations,

$$\begin{aligned}
\mathbf{P}(\|Y^*\|^2 \geq \beta d/n) &= \mathbf{P}(n(X_1^2 + \cdots + X_d^2) \geq d\beta(X_1^2 + \cdots + X_n^2)) \\
&= (1+2td\beta)^{-(n-d)/2}(1+2t(d\beta-n))^{-d/2}
\end{aligned}$$

for $0 < t < \frac{1}{2}(n - d\beta)$. This time use $t = -t_0$. ▮

Lemma 8 *Fix any $\delta, \epsilon \in (0,1)$. Let u, v be any two points in $\mathbb{R}^n$. Suppose these points are projected into a randomly chosen subspace of dimension $d < n$, where*

$$d \ \geq \ \frac{2}{\epsilon^2/2 - \epsilon^3/3}\ln\frac{2}{\delta}.$$

If u^*, v^* denote the projected points, then with probability at least $1-\delta$ over the choice of projection,

$$(1-\epsilon)\,\|u-v\|^2\,\frac{d}{n} \ \leq \ \|u^* - v^*\|^2 \ \leq \ (1+\epsilon)\,\|u-v\|^2\,\frac{d}{n}.$$

Proof. Without loss of generality $\|u - v\| = 1$. Then $\|u^* - v^*\|$ has the same distribution as $\|Y^*\|$ in Lemma 7. Applying that lemma with $\beta = 1 - \epsilon$ and $\beta = 1 + \epsilon$ and using the inequalities $\ln(1 - x) \leq -x - x^2/2$ (for $x \geq 0$) and $\ln(1 + x) \leq x - x^2/2 + x^3/3$ (for $x \geq 0$), we get

$$\begin{aligned}\mathbf{P}(\|u^* - v^*\|^2 \leq (1-\epsilon)d/n) &\leq \exp(-d\epsilon^2/4),\\ \mathbf{P}(\|u^* - v^*\|^2 \geq (1+\epsilon)d/n) &\leq \exp(-d(\epsilon^2/2 - \epsilon^3/3)/2),\end{aligned}$$

and the sum of these failure probabilities is at most δ. ▮

The Johnson-Lindenstrauss lemma follows immediately by applying Lemma 8 to all $\binom{m}{2}$ pairs of distinct points and then taking a union bound.

Literature cited

Acid, S., de Campos, L.M., González, A., Molina, R. & Pérez de la Blanca, N. (1991) Learning with CASTLE. In *Symbolic and Quantitative Approaches to Uncertainty*, Lecture Notes in Computer Science 548, 99-106. Berlin: Springer-Verlag.

Alimonti, P. & Kann, V. (1997) Hardness of approximating problems on cubic graphs. *Third Italian Conference on Algorithms and Complexity*, Lecture Notes in Computer Science 1203, Berlin: Springer.

Arora, S. & Lund, C. (1996) Hardness of approximations. In *Approximation algorithms for NP-hard problems*, ed. D. Hochbaum. Boston: PWS.

Bellman, R.E. (1961) *Adaptive control processes: a guided tour.* Princeton: Princeton University Press.

Basu, S. & Micchelli, C.A. (1998) Parametric density estimation for the classification of acoustic feature vectors in speech recognition. *Nonlinear Modeling*, eds. J. Suykens and J. Vandewalle. Boston: Kluwer.

Cheeseman, P. & Stutz, J. (1995) Bayesian Classification (AutoClass): Theory and Results. *Advances in Knowledge Discovery and Data Mining*, eds. U. Fayyad, G. Piatetsky-Shapiro, P. Smyth and & R. Uthurusamy. Menlo Park, CA: AAAI Press.

Chickering, D.M. (1996) Learning Bayesian networks is NP-complete. *Learning from data: AI and Statistics V*, 121-130. New York: Springer.

Chow, C.K. & Liu, C.N. (1968) Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14:462-467.

Cooper, G. (1990) The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, 42:393-405.

Cover, T.M. & Thomas, J.A. (1991) *Elements of information theory.* New York: John Wiley.

Dagum, P. & Luby, M. (1993) Approximate probabilistic reasoning in Bayesian belief networks is NP-hard. *Artificial Intelligence*, 60:141-153.

Dasgupta, S. (1997) The sample complexity of learning fixed-structure Bayesian networks. *Machine Learning*, 29:165-180. Boston: Kluwer.

Dasgupta, S. & Gupta, A. (1999) An elementary proof of the Johnson-Lindenstrauss lemma. Technical Report 99-006, International Computer Science Institute, Berkeley.

Dempster, A.P., Laird, N.M. & Rubin, D.B. (1977) Maximum-likelihood from incomplete data via the EM algorithm. *J. Royal Statist. Soc. Ser. B*, 39:1-38.

Diaconis, P. & Freedman, D. (1984) Asymptotics of graphical projection pursuit. *Annals of Statistics*, 12:793-815.

Duda, R.O. & Hart, P.E. (1973) *Pattern Classification and Scene Analysis.* New York: John Wiley.

Dudley, R.M. (1979) Balls in R^k do not cut all subsets of $k + 2$ points. *Advances in Mathematics*, 31:306-308.

Durrett, R. (1996) *Probability: Theory and Examples.* Belmont, CA: Duxbury.

Edmonds, J. (1967) Optimum branchings. *Journal of Research of the National Bureau of Standards*, 71B(4):233-240.

Feder, T. & Greene, D. (1988) Optimal algorithms for approximate clustering. *ACM Symposium on Theory of Computing.*

Feller, W. (1966) *An Introduction to Probability Theory and its Applications*, vol. II. New York: Wiley.

Frankl, P. & Maehara, H. (1988) The Johnson-Lindenstrauss lemma and the sphericity of some graphs. *Journal of Combinatorial Theory Ser. B*, 44:355-365.

Friedman, N. (1997) Learning Bayesian networks in the presence of missing values and hidden variables. *International Conference on Machine Learning.*

Freund, Y. & Mansour, Y. (1999) Estimating a mixture of two product distributions. *ACM Conference on Computational Learning Theory.*

Geiger, D., Paz, A. & Pearl, J. (1990) Learning causal trees fron dependence information. *Proceedings of the Eighth National Conference on Artificial Intelligence*, 770–776. Cambridge: AAAI Press.

González, T.F. (1985) Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38:293-306. North-Holland.

Gupta, A. (1999) Embedding tree metrics into low dimensional Euclidean spaces. *ACM*

Symposium on Theory of Computing.

Haussler, D. (1992) Decision-theoretic generalizations of the PAC model for neural net and other learning applications. *Information and Computation*, 100:78-150.

Hochbaum, D.S. & Shmoys, D.B. (1985) A best possible heuristic for the k-center problem. *Mathematics of Operations Research*, 10(2):180-184.

Höffgen, K.-U. (1993) Learning and robust learning of product distributions. *Proceedings of the Sixth Annual ACM Conference on Computational Learning Theory*, 77-83.

Horn, R.A. & Johnson, C.R. (1985) *Matrix Analysis.* Cambridge University Press.

Jaakkola, T. & Jordan, M. (1999) Variational probabilistic inference and the QMR-DT network. *Journal of Artificial Intelligence Research*, 1:1-15.

Jelinek, F. (1997) *Statistical methods for speech recognition.* Cambridge: MIT Press.

Johnson, W.B. & Lindenstrauss, J. (1984) Extensions of Lipschitz mapping into Hilbert space. *Contemp. Math.*, 26:189-206.

Kaelbling, L.P. & Ortiz, L. (1999) Accelerating EM: an empirical study. *Uncertainty in Artificial Intelligence.*

Kearns, M., Mansour, Y., Ron, D., Rubinfeld, R., Schapire, R. & Sellie, L. (1994) On the learnability of discrete distributions. *ACM Symposium on Theory of Computing.*

Kearns, M.J. & Vazirani, U.V. (1994) *An Introduction to Computational Learning Theory.* Cambridge: MIT Press.

Kettenring, J. & Pregibon, D., eds. (1996) *Statistics and Massive Data Sets*, Report to the Committee on Applied and Theoretical Statistics, National Research Council, Washington, DC.

Lauritzen, S. (1996) *Graphical models.* Oxford: Oxford University Press.

Lauritzen, S. & Spiegelhalter, D. (1988) Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society, Series B*, 50:157-224.

Lewis, D.D. (1996) Challenges in machine learning for text classification. *Proceedings of the Ninth Annual Conference on Computational Learning Theory*, New York: ACM Press. David Lewis' homepage, www.research.att.com/˜lewis, can be consulted for details on the Reuters data set.

Lindsay, B. (1995) *Mixture Models: Theory, Geometry, and Applications.* American Statistical Association, Virginia.

Longsworth, L.G. (1942) Recent advances in the study of protein by electrophoresis. *Chem-*

ical Review, 30:323-340.

Meilă, M. & Heckerman, D. (1998) An experimental comparison of several clustering methods. Microsoft research technical report MSR-TR-98-06.

Motwani, R. & Raghavan, P. (1995) *Randomized algorithms.* New York: Cambridge University Press.

Pach, J. & Agarwal, P. (1995) *Combinatorial Geometry.* Wiley.

Papadimitriou, C. & Yannakakis, M. (1991) Optimization, approximation, and complexity classes. *Journal of Computer and System Sciences*, 43:425-440.

Pearl, J. (1988) *Probabilistic reasoning in intelligent systems.* San Mateo, CA: Morgan Kaufmann.

Pearson, K. (1894) Contributions to the mathematical theory of evolution. *Phil. Trans. Royal Soc. Ser. A*, 185:71-110.

Petrov, V.V. (1995) Limit Theorems of Probability Theory. New York: Oxford University Press.

Rabiner, L. (1989) A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257-285.

Redner, R.A. & Walker, H.F. (1984) Mixture densities, maximum likelihood and the EM algorithm. *SIAM Review*, 26(2):195-239.

Russell, S., Binder, J., Koller, D. & Kanazawa, K. (1995) Local learning in probabilistic networks with hidden variables. *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence.* Los Altos, CA: William Kaufmann.

Seymour, P.D. (1995) Packing directed circuits fractionally. *Combinatorica*, 15(2):281-288.

Shwe, M., Middleton, B., Heckerman, D., Henrion, M., Horvitz, E., Lehmann, H. & Cooper, G. (1991) Probabilistic diagnosis using a reformulation of the INTERNIST-1/QMR knowledge base I. *Methods of Information in Medicine*, 30:241-255.

Titterington, D.M., Smith, A.F.M. & Makov, U.E. (1985) *Statistical Analysis of Finite Mixture Distributions.* London: John Wiley.

Valiant, L. (1984) A theory of the learnable. *Communications of the ACM*, 27:1134-1142.

Xu, L. & Jordan, M. (1996) On convergence properties of the EM algorithm for Gaussian mixtures. *Neural computation*, 8:129-151.