A search method for Gaussian mixtures using MML

Elsevier[1]

Radarweg 29, Amsterdam

Elsevier Inc[a,b], Global Customer Service[b,]*

[a]1600 John F Kennedy Boulevard, Philadelphia
[b]360 Park Avenue South, New York

Abstract

Keywords: Search, mml

1. Introduction

2. MML

A complex model is not an optimal one, unless its complexity is justifiable by the added explanatory power. There is an established connection between simplicity and truth, as beautifully elucidated by the Occam's Razor principle. MML is a statistical technique that formalizes and quantifies this principle.

The basic idea is to consider the transmission of a set of data and a model that describes it. Clearly one can make increasingly complex models to obtain better fits – think of epicycles in the solar system. Suppose we try to best represent N data points. Traditional curve fitting methods would give a polynomial of degree N-1, which is perhaps the most precise, but it is also the most complex, and the one that most models the noise, by spuriously *over-fitting non-existing patterns.* Over-fitting can result in a model that may be far from the truth. MML will only consider a more complex model if the encoding of the complex model and the data is more efficient than the less complex model, i.e. it finds the minimum message length needed to transmit the data and the model. It is this *true* model that MML aims to find.

It differs from PCA in that it does find groups (or *classes* in the notation of MML). For each class it can provide a "latent factor" which is akin to the first PC in a PCA study. Let us be a little more quantitative.

The MML principle states that the best explanation of the data is the one that leads to the shortest message [1], where a *message* takes into account both

*Corresponding author
Email address: support@elsevier.com (Global Customer Service)
URL: www.elsevier.com (Elsevier Inc)
[1]Since 1880.

the complexity of the model, and its explanatory power. In essence, MML infers the most concise way of encoding the model, θ, and the data, x, given the model, $x|\theta$. A complex model will produce a longer first part of the message than a simple model, since more quantities must be stated. On the other hand, the length of the second part of the message decreases with more accurate and complex models, since less data has to be described fully. The *message* must encode two parts: the model, and the data. The encoding of the message is based on Shannon's information theory. According to Shannon, the information we gain from an event e is $h(e) = -\log_2 p(e)$, where $p(e)$ is the probability of that event. The information content is biggest for the improbable outcomes, and smallest for outcomes that we are almost certain about. An outcome that has a probability close to one has close to zero information content, since we don't learn anything new from it, whereas the rarer events have much higher information content.

Consider the simplified illustrative example of rolling an unbiased four-sided die (d4), where each of the four outcomes has equal probability $p_i = 0.25$. The information content in each datum is $-\log_2 0.25 = 2$. In total $\sum_{i=1}^{4} -\log_2(p_i) = 2 + 2 + 2 + 2 = 8$ bits are required to describe the four outcomes. The model description could be $\{00, 01, 10, 11\}$: this is the first part of the message.

Now assume that the die is biased, with probabilities $p(1) = 0.5, p(2) = 0.25, p(3) = 0.125$, and $p(4) = 0.125$. In this case, the number of bits required to represent each alternative is different. One bit is required for 1, since $-\log_2 0.5 = 1$, and the codeword could be the single binary value 1. Similarly, 2 bits are required for the outcome 2, which can be represented with codeword 01. It follows that possible codewords for the remaining alternatives, all using the bit "1" to indicate the end of each datum, are 001 for 3, 0001 for 4, 00001 for 5, and 000001 for six. The length of the model required to describe the outcomes of the biased die would be $\sum_{i=1}^{4} -\log_2(p_i) = 1 + 2 + 3 + 3 = 9$ bits, and the codewords would be $\{0, 10, 110, 111\}$. The second description is 1 bit longer than the first, but, since the most frequent events have a shorter codeword, the second part of the message (the data given the model) is shorter. The overall message length of the second description is shorter than the first, when applied to the biased die.

To illustrate this point let's encode 1000 outcomes of a biased die, where 500 outcomes are 1, 250 are 2, 125 are 3, and 125 are 4. With the first model each outcome is encoded with two bits, so the length of the second part of the message is $1000 \times 2 = 2000$. The total message length is the sum of the model length and the length of the data given the model, or 2008 bits. But if we encode the data with the second model, the length of the data given the model is $500 + (250 \times 2) + (125 \times 3) + (125 \times 3) = 1750$. The total length is $1750 + 9 = 1759$. Thus the second model, although it is more complex (takes more bits to describe), creates a shorter encoded message, and hence is a better description than the shorter model.

Formally, the length of the first part of the message is $-\log_2 h(\theta)$, and that of the second part is $-\log_2 f(x|\theta)$, where $h(\theta)$ is the prior probability

of model θ, and $f(x|\theta)$ is the likelihood of data x assuming model θ. The total message length is the sum of the two parts, i.e. $-\log_2 h(\theta) - \log_2 f(x|\theta)$, which is minimised precisely when $h(\theta)f(x|\theta)$ is maximised. The multiplication of the prior and the likelihood is proportional to the posterior *probability* $g(\theta|x)$. Thus the MML estimate $h(\theta)f(x|\theta)$ is equal to the posterior mode. This is similar in spirit to the (Bayesian) Maximum A Posteriori (MAP) posterior mode, but with the benefit of statistical invariance under re-parameterisation [2].

3. MML for clustering with a single latent factor model

MML is used to identify groups (also known as "classes") of stars with the same composition, and determine the number of independent dimensions in $\mathcal{C}$-space . It is important to note that MML can answer these two questions simultaneously. Without MML one would use some clustering technique to find the groups of similar stars, and then use a dimensionality reduction method like PCA to find the underlying patterns of nucleosynthesis within these groups. MML performs these two tasks simultaneously. The latent factors within MML are encoded into the optimisation when finding the groups. As a result, MML is able to assess whether adding an extra group hinders the fit of the latent factor model. In current (non-MML) methods these two tasks are not performed jointly, which may lead to groups that are a perfect fit, but with a biased latent factor model. MML will be including the underlying nuclear physics when finding the groups within the data (of course, we can instruct it not to do this if we prefer). This is a significant advantage over other techniques that have been investigated so far.

For each group determined by MML, the latent factor is a vector that best encapsulates the location and distribution of this group in $\mathcal{C}$-space. Mixture modelling enables the partitioning of the data into overlapping groups. For each group, a single latent factor model is created. In MML we model this by constructing a two-part message, with the first part describing the mixture model (or hypothesis, θ) and the second part representing the encoded data (x) given the hypothesis. The assumed single latent factor model is $x_{nk} = \mu_k + \nu_n a_k + \sigma_k r_{nk}$, where μ_k is the estimated mean for each attribute or dimension k ($k = 1, \ldots, K$); $\vec{a} = (a_1, ..., a_K)$ is the factor load vector, giving the contribution of each variable to the underlying common factor, $\vec{\nu} = (\nu_1, ..., \nu_N)$ is the factor score vector, representing how much each data item in turn has of the factor; σ_k is the standard deviation in attribute k, and the variates r_{nk} are independently and identically distributed in $N(0, 1)$. In essence, the model regards each observation x_{nk} as determined by the mean μ_k, depending on an unobserved variable ν_n and related to a specific variability with standard deviation σ_k. The MML *message* contains the following components:

1a) A statement of the number of groups, or classes, C. Each class is assumed to have the same prior probability, equal to 2^{-K}. The length of this part is equal to the negative logarithm of the prior, i.e. K.

1b) The relative abundances (or mixing proportions) of the amount(s) of data in each group. The length of this part is equal (within a small constant) to the negative log of the multiplication of the prior probabilities of each group ($h(\vec{p})$) divided by the square root of the Fisher information ($\sqrt{F(\vec{p_k})}$). The Fisher information matrix contains the expected second-order partial derivatives of the log-likelihood function, and quantifies how sharply the likelihood function peaks, and as a result how precisely the parameters should be stated.

In the case of uniform prior probabilities $h(\vec{p}) = (K-1)!$, and the Fisher information $F(\vec{p}) = N^{K-1}/(\prod_{k=1}^{K} p_k)$. It follows that the length of this part is equal to $-\log(h(\vec{p}) \times (1/\sqrt{\kappa_K{}^K F(\vec{p})})) = -\log((K-1)!) + (K/2)\log(\kappa_K) + ((K-1)/2)\log(N) - (1/2)\sum_{k=1}^{K}\log(p_k)$, where κ_K is a lattice constant (varying between $1/12$ and $1/(2\pi e)$).

1c) The distributional parameters μ_k and σ_k (one for each attribute or dimension k) and the factor load (and score) vectors for each group, again rounded off as above using the Fisher information.

The explanation length of this part is $(N-1)\sum_k log\sigma_k + \frac{1}{2}[Klog(N\sum_n \nu_n^2 - (\sum_n \nu_n)^2) + (N+K-1)log(1+\sum_k (a_k/\sigma_k)^2)] + \frac{1}{2}[\sum_n \nu_n^2 + \sum_n \sum_k (x_{nk}\mu_k - v_n a_k)^2/\sigma_k^2]$. This part measures the information required to encode the latent factor model for each class. The more classes, the longer the message length is.

2) The data given the model, whose cost is the well-known statistical log-likelihood. In the case of a normal distribution, this is equal to

$$-\log p_k(x)(1/(\sigma_k\sqrt{(2\pi)}e^{-\frac{(x-\mu_k)^2}{2\sigma_k^2}}).$$

Parts 1a and 1b encode the parameters from the clustering process, part 1c estimates the complexity of the latent factor model, and part 2 encodes the likelihood which represents the goodness-of-fit. The estimation of the multinomial parameters from part 1b and the distributional parameters from part 1c is done by quantifying the trade-off between stating the parameters imprecisely, so as to have higher prior probability in the relevant uncertainty region and keep the first part of the message short, and the goodness of fit to the data, which is typically better if we state the parameter estimates more precisely. Quantitatively, this is done by minimising the sum of the lengths of the four components (1a, 1b, 1c and 2 above). Single latent factor analysis attempts to reduce the data to one dimension. If the factor is very weak, MML may decide that the cost of describing the factor in part 1c is greater than the saving that would occur in part 2. In such a case MML will posit that there is no factor; i.e. including this factor would be over-fitting the data, taking us away from the optimum model.

Similarly, if MML finds that the data for a specific element do not affect the selection of groups, then it will *ignore that element.* This could tell us something important - such as that there is a large random component for that element

or that there are significant observational errors ruining the usefulness of that measurement. These are significant insights into the data that are, importantly, quantified by the MML method.

4. Message Length

The MML principle requires that we encode everything required to reconstruct the message, including our prior beliefs on the model parameters. Consider that we have N data points each with D dimensions which are to be modelled by a finite number of Gaussian mixtures. Therefore the message for a finite number of Gaussian mixtures must encode:

1. The number of Gaussian mixtures, K.
2. The relative weights $\mathbf{w}$ of the $K-1$ Gaussian mixtures. Only $K-1$ weights must be encoded because $\sum_{k=1}^{K} w_k = 1$.
3. The component parameters $\boldsymbol{\mu}$ and $\mathbf{C}$ for all K Gaussian mixtures.
4. The data, given the model parameters.
5. The lattice quantisation constant $\kappa(Q)$ for the number of model parameters Q.

We provide the message length of each component in turn before outlining our full objective function. We assume a prior on the number of mixtures K of $p(K) \propto 2^{-K}$ [?] such that the length of the optimal lossless message to encode K is

$$I(K) = -\log p(K) = K \log 2 + \text{constant}. \tag{1}$$

We assume a uniform prior on the individual weights w_k. The weights $\mathbf{w}$ are drawn from a multinomial distribution such that they can be optimally encoded in a message with length[?]:

$$I(\mathbf{w}) = \frac{K-1}{2} \log N - \frac{1}{2} \sum_{k=1}^{K} \log w_k - \log \Gamma(K) \quad . \tag{2}$$

In order to properly encode the means $\boldsymbol{\mu}$ and covariance matrices $\mathbf{C}$ for all K mixtures, we must encode both our prior belief on those parameters and the determinant of the expected Fisher information matrix. For the k-th mixture the joint message length is

$$\begin{aligned} I(\boldsymbol{\mu}_k, \mathbf{C}_k) &= -\log \left(\frac{p(\boldsymbol{\mu}_k, \mathbf{C}_k)}{\sqrt{|\mathcal{F}(\boldsymbol{\mu}_k, \mathbf{C}_k)|}} \right) \\ I(\boldsymbol{\mu}_k, \mathbf{C}_k) &= -\log p(\boldsymbol{\mu}_k, \mathbf{C}_k) + \frac{1}{2} \log |\mathcal{F}(\boldsymbol{\mu}_k, \mathbf{C}_k)| \end{aligned} \tag{3}$$

and for all mixtures:

$$I(\boldsymbol{\mu}, \mathbf{C}) = -\sum_{k=1}^{K} \log p(\boldsymbol{\mu}_k, \mathbf{C}_k) + \frac{1}{2} \sum_{k=1}^{K} \log |\mathcal{F}(\boldsymbol{\mu}_k, \mathbf{C}_k)| \quad . \tag{4}$$

We use an improper uniform prior on the mean $\boldsymbol{\mu}$ for all mixtures such that $\mathcal{U}(\boldsymbol{\mu}) = (-\infty, +\infty)$. However, the MML principle demands proper priors. In practice we make this prior proper by assuming the bounds on $p(\boldsymbol{\mu})$ are very large, but we do so without actually specifying the value of the bounds because they add only a constant value to the total message length[**?**]. We adopt a conjugate inverted Wishart prior for the covariance matrix of an individual mixture [e.g., Section 5.2.3. of **?**]:

$$p(\boldsymbol{\mu}_k, \mathbf{C}_k) \propto |\mathbf{C}_k|^{\frac{D+1}{2}} \quad . \tag{5}$$

Following Eq. 4, we require the logarithm of the determinant of the expected Fisher information matrix for $\boldsymbol{\mu}_k$ and $\mathbf{C}_k$ in order to encode the full message length. We approximate the determinant of the Fisher information $|\mathcal{F}(\boldsymbol{\mu}_k, \mathbf{C}_k)|$ as the product of $|\mathcal{F}(\boldsymbol{\mu}_k)|$ and $|\mathcal{F}(\mathbf{C}_k)|$ [**?** **?**]. In order to derive $|\mathcal{F}(\boldsymbol{\mu}_k)|$ we take the second derivative of the log-likelihood function $\mathcal{L}(\mathbf{y}|\boldsymbol{\mu}, \mathbf{C})$, which yields:

$$|\mathcal{F}(\boldsymbol{\mu}_k)| = (N w_k)^D |\mathbf{C}_k|^{-1} \quad . \tag{6}$$

We make use of an analytical expression for $|\mathcal{F}(\mathbf{C}_k)|$ [**?** **?**]:

$$|\mathcal{F}(\mathbf{C}_k)| = (N w_k)^{\frac{D(D+1)}{2}} 2^{-D} |\mathbf{C}_k|^{-(D+1)} \quad . \tag{7}$$

Combining these expressions gives our approximation for the determinant of the expected Fisher information $|\mathcal{F}(\boldsymbol{\mu}_k, \mathbf{C}_k)|$:

$$\begin{aligned}
|\mathcal{F}(\boldsymbol{\mu}_k, \mathbf{C}_k)| &\approx |\mathcal{F}(\boldsymbol{\mu}_k)| \cdot |\mathcal{F}(\mathbf{C}_k)| \\
|\mathcal{F}(\boldsymbol{\mu}_k, \mathbf{C}_k)| &\approx (N w_k)^D |\mathbf{C}_k|^{-1} (N w_k)^{\frac{D(D+1)}{2}} 2^{-D} |\mathbf{C}_k|^{-(D+1)} \\
|\mathcal{F}(\boldsymbol{\mu}_k, \mathbf{C}_k)| &\approx (N w_k)^{\frac{D(D+3)}{2}} 2^{-D} |\mathbf{C}_k|^{-(D+2)} \quad .
\end{aligned} \tag{8}$$

Substituting Eqs. 5 and 8 into Eq. 4 gives the message length to encode the component parameters $\boldsymbol{\mu}$ and $\mathbf{C}$ for all K mixtures:

$$\begin{aligned}
I(\boldsymbol{\mu}, \mathbf{C}) &= -\sum_{k=1}^{K} \log p(\boldsymbol{\mu}_k, \mathbf{C}_k) + \frac{1}{2} \sum_{k=1}^{K} \log |\mathcal{F}(\boldsymbol{\mu}_k, \mathbf{C}_k)| \\
I(\boldsymbol{\mu}, \mathbf{C}) &= -\sum_{k=1}^{K} \log |\mathbf{C}_k|^{\frac{D+1}{2}} + \frac{1}{2} \sum_{k=1}^{K} \log \left[(N w_k)^{\frac{D(D+3)}{2}} 2^{-D} |\mathbf{C}_k|^{-(D+2)} \right] \\
I(\boldsymbol{\mu}, \mathbf{C}) &= \frac{D(D+3)}{4} \sum_{k=1}^{K} \log N w_k - \frac{(D+1)}{2} \sum_{k=1}^{K} \log |\mathbf{C}_k| - \frac{KD}{2} \log 2 \\
&\quad - \frac{D+2}{2} \sum_{k=1}^{K} \log |\mathbf{C}_k| \\
I(\boldsymbol{\mu}, \mathbf{C}) &= \frac{D(D+3)}{4} \sum_{k=1}^{K} \log N w_k - \frac{2D+3}{2} \sum_{k=1}^{K} \log |\mathbf{C}_k| - \frac{KD}{2} \log 2
\end{aligned} \tag{9}$$

If we assume that the data have homoskedastic noise properties, then the precision of each measurement $\mathcal{E}$ relates the probability of a datum $p(y_n)$ can be related to the probability density of the datum, given the model $p(y_n) = \mathcal{E}^D p(y_i|\mathcal{M})$. In this work we adopt $\mathcal{E} = 0.01$, but we note that the value of $\mathcal{E}$ has no effect on our inferences. The message length of encoding a datum given the model is,

$$I(\mathbf{y}_n) = -\log p(\mathbf{y}_n) = -D \log \mathcal{E} - \log \sum_{k=1}^{K} w_k f_k(\mathbf{y}_n|\boldsymbol{\mu}_k, \mathbf{C}_k) \tag{10}$$

and for the entire data:

$$I(\mathbf{y}|\boldsymbol{\theta}) \quad = \quad -ND \log \mathcal{E} - \sum_{n=1}^{N} \log \sum_{k=1}^{K} w_k f_k(\mathbf{y}_n|\boldsymbol{\mu}_k, \mathbf{C}_k) \quad . \tag{11}$$

The final part of the message is to encode the lattice quantisation constant, which arises from the approximation to the (so-called) strict MML, where parameters are quantised into intervals in high dimensional space in order to be losslessly encoded. We use an approximation of the lattice constant [see Sections 5.1.12 and 3.3.4 of **?**] such that,

$$\log \kappa(Q) = \frac{\log Q\pi}{Q} - \log 2\pi - 1 \quad , \tag{12}$$

where Q is the total number of free parameters in the model. Each Gaussian component in the mixture requires D parameters to specify $\boldsymbol{\mu}$, and a covariance matrix with off-diagonal terms requires $\frac{1}{2}D(D+1)$ parameters. Because only $K-1$ weights need to be encoded, the total number of free parameters is

$$Q = K\left[\frac{D(D+3)}{2} + 1\right] - 1 \quad . \tag{13}$$

Each k-th mixture is a D-dimensional Gaussian that contributes a relative weight w_k such that $\sum_{k=1}^{K} w_k = 1$. We will assume that the data have homoscedastic noise properties.

The data have the same error value, y_{err}, in all D dimensions, for all N observations.

The message length for the data is given by:

Here we introduce the individual components that contribute to the message length.

Combining these expressions, the complete message length for K mixtures is given by

$$I_K \quad = \quad K \log 2 + \frac{(K-1)}{2} \log N - \frac{1}{2} \sum_{k=1}^{K} \log w_k - \log |\Gamma(K)| + \mathcal{L}(\mathbf{y}|\boldsymbol{\theta}) - DN \log y_{err} \tag{14}$$

$$+ \quad \frac{1}{2} \sum_{k=1}^{K} \left[\frac{D(D+3)}{2} \log N w_k - (D+2) \log |\mathbf{C}_k| - D \log 2 \right] - \frac{Q}{2} \log(2\pi) + \frac{\log Q\pi}{2} \tag{15}$$

where $Q = \frac{1}{2}DK(D+3) + K - 1$, the number of free parameters, and $\mathcal{L}$ is the log-likelihood of a multivariate gaussian distribution.

Say we wanted to calculate whether another mixture was warranted. If another mixture were preferred then we would want:

$$\Delta I_{K+1} - I_K < 0 \tag{16}$$

The expression is given as:

$$\begin{aligned}
\Delta I_{K+1} - I_K &= (K+1)\log 2 - K\log 2 && (17)\\
&+\frac{(K)}{2}\log N - \frac{1}{2}\sum_{k=1}^{K+1}\log {w_k}^{(new)} - \log|\Gamma(K+1)| && (18)\\
&-\frac{(K-1)}{2}\log N + \frac{1}{2}\sum_{k=1}^{K}\log w_k + \log|\Gamma(K)| && (19)\\
&+\mathcal{L}^{(new)} - DN\log y_{err} && (20)\\
&-\mathcal{L}^{(old)} + DN\log y_{err} && (21)\\
&+\frac{1}{2}\sum_{k=1}^{K+1(new)}\left[\frac{D(D+3)}{2}\log N w_k - (D+2)\log|\mathbf{C}_k| - D\log 2\right] && (22)\\
&-\frac{1}{2}\sum_{k=1}^{K(old)}\left[\frac{D(D+3)}{2}\log N w_k - (D+2)\log|\mathbf{C}_k| - D\log 2\right] && (23)\\
&-\frac{Q^{(new)}}{2}\log(2\pi) + \frac{\log Q^{(new)}\pi}{2} && (24)\\
&+\frac{Q^{(old)}}{2}\log(2\pi) - \frac{\log Q^{(old)}\pi}{2} && (25)
\end{aligned}$$

By making use of $\log\Gamma(K) - \log\Gamma(K+1) = -\log K$ and re-arranging the expression:

$$\begin{aligned}
\Delta I_{K+1} - I_K &= \log 2 + \frac{1}{2}\log N - \log K - \frac{1}{2}\left(\sum_{k=1}^{K+1}\log w_k^{(new)} - \sum_{k=1}^{K}\log w_k^{(old)}\right)\\
&+\mathcal{L}^{(new)} - \mathcal{L}^{(old)}\\
&+\frac{1}{2}\left[\frac{D(D+3)}{2}\left(\sum_{k=1}^{K+1}\log N w_k^{(new)} - \sum_{k=1}^{K+1}\log N w_k^{(old)}\right) - (D+2)\left(\sum_{k=1}^{K+1}\log|\mathbf{C}_k|^{(new)} - \sum_{k=1}^{K+1}\right.\right.\\
&+\frac{\log(2\pi)}{2}(Q^{(old)} - Q^{(new)}) + \frac{\pi}{2}\left(\log Q^{(new)} - \log Q^{(old)}\right)
\end{aligned}$$

Expanding the Q terms:

$$
\begin{aligned}
Q^{(old)} - Q^{(new)} &= \frac{1}{2}DK(D+3) + K - 1 - \frac{1}{2}D(K+1)(D+3) + (K+1) - 1 \\
Q^{(old)} - Q^{(new)} &= -\frac{1}{2}D(D+3) + 2K - 1 \qquad (27)
\end{aligned}
$$

And making use of the following logarithmic identities,

$$
\begin{aligned}
\log Q^{(new)} &= \log\left(\frac{1}{2}D(K+1)(D+3) + K\right) \\
&= \log\left(\frac{1}{2}D(K+1)(D+3)\right) + \log\left(1 + \frac{K}{\frac{1}{2}D(K+1)(D+3)}\right) \qquad (28) \\
\log Q^{(old)} &= \log\left(\frac{1}{2}DK(D+3) + K - 1\right) \\
&= \log\left(\frac{1}{2}DK(D+3)\right) + \log\left(1 + \frac{K-1}{\frac{1}{2}DK(D+3)}\right) \qquad (29)
\end{aligned}
$$

gives us,

$$
\begin{aligned}
\log Q^{(new)} - \log Q^{(old)} &= \log\left(\tfrac{1}{2}D(K+1)(D+3)\right) - \log\left(\tfrac{1}{2}DK(D+3)\right) \\
&\quad + \log\left(1 + \tfrac{K}{\frac{1}{2}D(K+1)(D+3)}\right) - \log\left(1 + \tfrac{K-1}{\frac{1}{2}DK(D+3)}\right). \qquad (30)
\end{aligned}
$$

The second row of terms can be ignored because they are very small (typically less than 1 bit). This is because as $K \to \infty$, $2K/D(K+1)(D+3) \to 1$, thus $\log\left(1 + \frac{K}{\frac{1}{2}D(K+1)(D+3)}\right) \to \log 2$. Similarly as $D \to \infty$, $2K/D(K+1)(D+3) \to 0$.

As $K \to \infty$ then $2(K-1)/DK(D+3) \to 1$ and as $D \to \infty$ then $2(K-1)/DK(D+3) \to 0$ and thus $\log\left(1 + \frac{K-1}{\frac{1}{2}DK(D+3)}\right) \approx 0$.
Ignoring these minor terms:

$$
\begin{aligned}
\log Q^{(new)} - \log Q^{(old)} &\approx \log\left(\frac{1}{2}D(K+1)(D+3)\right) - \log\left(\frac{1}{2}DK(D+3)\right) \\
\log Q^{(new)} - \log Q^{(old)} &\approx \log(K+1) - \log K \qquad (31)
\end{aligned}
$$

Substituting Eqs. 31 and 27 into 26 yields:

$$
\begin{aligned}
\Delta I_{K+1} - I_K &\approx \log 2 + \frac{1}{2}\log N - \log K - \frac{1}{2}\left(\sum_{k=1}^{K+1}\log w_k^{(new)} - \sum_{k=1}^{K}\log w_k^{(old)}\right) \\
&\quad + \mathcal{L}^{(new)} - \mathcal{L}^{(old)} \\
&\quad + \frac{1}{2}\left[\frac{D(D+3)}{2}\left(\sum_{k=1}^{K+1}\log N w_k^{(new)} - \sum_{k=1}^{K+1}\log N w_k^{(old)}\right) - (D+2)\left(\sum_{k=1}^{K+1}\log|\mathbf{C}_k|^{(new)} - \sum_{k=1}^{K+1}\log|\mathbf{C}\right.\right. \\
&\quad + \frac{\log(2\pi)}{2}(-\frac{1}{2}D(D+3) + 2K - 1) + \frac{\pi}{2}\left(\log(K+1) - \log K\right)
\end{aligned}
$$

5. Related Work in Inferring the Number of Mixture Components

Akaike information criterion (AIC) [3] chooses a model that minimizes the Kullback-Leibler distance between the model and the truth, defined as:

AIC = -2 (ln (likelihood)) + 2 K

where likelihood is the probability of the data given a model and K is the number of free parameters in the model.

BIC

6. The k-means++ algorithm

The k-means++ algorithm extends the k-means method with an efficient way of choosing centres. Let $D(x)$ denote the shortest distance from a data point x to the closest centre that has already been selected. The k-means++ has the following steps:

1. Selects a centre c_1 uniformly at random from the N data points.
2. Selects a new centre c_i from the data points with probability $\frac{D(x)^2}{\sum_{x \in X}} D(x)^2$
3. Repeat Step 2. until k centres have been selected.
4. Proceed as with the standard k-means algorithm.

k-means++ is O(log k)-Competitive.

7. The Heuristic

Re the case of two (or more) classes with the same mean: In 2D, a simple case in point is two classes with the same mean such that $\sigma_{1,x} = \sigma_{2,y} >> \sigma_{2,x} = \sigma_{1,y}$. Andy said that this and higher dimensional stuff can be picked up by dropping to 1 dimension along the eigenvector of largest eigenvalue.

I wonder also about the merits of dropping to 2 dimensions (if we have at least 2 attributes), being the eigenvectors of the two largest eigenvalues.

One could possibly check whether uniform radially directly from the original 2D projection without having to project it to a circle (i.e., it might be pretty easy to work out what the angle of each point would be post-transformation without having to do the transformation). The 2D projection is 1D in the sense that we only care about the angle around the circle. And then some test (e.g., Kolmogorov-Smirnov test) for whether data is uniform in radial distribution. If not uniform, then perhaps grounds for splitting.

Re looking at things in 1 dimension (not 2D followed by 1D [0, 2 pi) angle as above but rather projecting on eigenvector of largest eigenvalue), we discussed moments - e.g., `https://en.wikipedia.org/wiki/Normal_distribution` .

Let's suppose we have a 1D component. We'll shift it (without loss of generality, w.l.o.g.) so that its mean is 0 and we'll shrink or expand it (again, w.l.o.g.) so that its s.d. (σ) is 1 (and variance v is 1).

We consider now splitting it into K components with mean 0, mixing weights (or proportions) $w_1, ..., w_i, ..., w_K$ and s.d.s σ_i (and variances $v_i = (\sigma_i)^2$). Let's set $K = 2$, so $w_2 = 1 - w_1$.

Suppose the moments of our current component are $1, 3\alpha_4, 15\alpha_6$ and $105\alpha_8, ...$ We get some equations - if we have the correct number of equations, they're simultaneous equations.

If we get too many equations (by using too many moments), then it becomes something of a fitting or regression problem to choose $w_1, ..., w_i, ..., w_K$ and s.d.s σ_i, variances $v_i = (\sigma_i)^2$.

Our simultaneous equations are:

$$\begin{aligned} w_1 v_1 + (1 - w_1) v_2 &= 1 \\ w_1 (v_1)^2 + (1 - w_1)(v_2)^2 &= 3\alpha_4 \\ w_1 (v_1)^3 + (1 - w_1)(v_2)^3 &= 15\alpha_6 \\ w_1 (v_1)^4 + (1 - w_1)(v_2)^4 &= 105\alpha_8 \end{aligned}$$

These might be messy to solve on the fly but one could pre-process by solving many of these beforehand (actually, approximately solving these beforehand, as it's only a heuristic, anyway) at the very start of the program while loading in the data.

So, for $1000 \times 1000 = 10^6$ different values of α_4 and α_6 or for $100 \times 100 \times 100 = 10^6$ different values of α_4, α_6 and α_8 one could pre-compute *approximate* 1,000,000ish 'solutions' to the above simultaneous equations.

Then, when running the program with real actual values of α_4 and α_6 (and α_8), one could just grab something from nearby in this pre-computed look-up table as putative values for w_1, v_1, v_2 (and etc., if need be).

[1] C. S. Wallace, Statistical and inductive inference by minimum message length, Springer Science & Business Media, 2005.

[2] D. L. Dowe, S. Gardner, G. Oppy, Bayes not bust! why simplicity is no problem for bayesians, The British Journal for the Philosophy of Science 58 (4) (2007) 709–754.

[3] H. Akaike, A new look at the statistical model identification, IEEE transactions on automatic control 19 (6) (1974) 716–723.