EECS E6720 Bayesian Models for Machine Learning

Columbia University, Fall 2016

Lecture 9, 11/10/2016

Instructor: John Paisley

Clustering with the Gaussian mixture model (GMM)

- We next look at a fundamental problem in machine learning—clustering data. There are many types of data we might want to cluster. We will focus on the Gaussian mixture model for data clustering, which restricts the data to be in $\mathbb{R}^d$.

- Data: That is, we're given a set of vectors $\{x_1, \dots, x_n\}$ where each $x_i \in \mathbb{R}^d$.

- Clustering: The main goal of clustering is to partition the data into groups (clusters) such that data within the same cluster are "similar" and data in different clusters are "different." Intuitively, we can think of the Gaussian mixture model as measuring closeness by distance, so each cluster defines a region in a space in the same way we might think of a region in a physical location.

- K-means is one way of clustering data, and arguable the most fundamental, which is why I bring it up here. It's not a probabilistic method and so we won't discuss it, but it's important to know. In this lecture, we will focus on the probabilistic approach to clustering using a Gaussian mixture model.

- Gaussian mixture models assume a probability density of the data that is a weighted sum of K Gaussian distributions. That is, we model the distribution of each data point $x_i \in \mathbb{R}^d$ as follows:

$$x_i \overset{iid}{\sim} p(x|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = \sum_{j=1}^{K} \pi_j \text{Normal}(x|\mu_j, \Lambda_j^{-1}) \tag{1}$$

- The number of Gaussians, K, is set in advance. There are ways to try to assess what it should be, but for now we just assume we have picked a setting. The model parameters are,
  - π : A probability distribution on Gaussians.
  - (μ_j, Λ_j) : The mean and precision of the jth Gaussian

- It might not be clear how to generate x_i from this distribution unlike a single Gaussian. Intuitively, you can think that, in a data set of size n, roughly π_j fraction of that data will come from the jth Gaussian with mean μ_j and precision (inverse covariance) Λ_j.

- For now, let's not think about priors on $\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}$. The likelihood of the data given $\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}$ is

$$p(x_1, \dots, x_n | \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) \;=\; \prod_{i=1}^{n} \left(\sum_{j=1}^{K} \pi_j \frac{|\Lambda_j|^{\frac{1}{2}}}{(2\pi)^{\frac{d}{2}}} \mathrm{e}^{-\frac{1}{2}(x_i-\mu_j)^T \Lambda_j (x_i-\mu_j)} \right) \tag{2}$$

- If we were to try to maximize this likelihood over $\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}$ (i.e., perform "maximum likelihood"), we would not make it very far before we realize that we can't do this simply and in closed form. That is, no derivative of this likelihood can be set to zero and solved for a parameter of interest. We therefore would need to resort to gradient methods if we want to directly maximize $\ln p(x_1, \dots, x_n | \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})$.

- However, recall that this is the setting where EM can help, and we will indeed see that that's the case for the GMM.

- Before discussing EM for the GMM, it's important here to point out that the EM algorithm does not equal "maximum likelihood" or "maximum a posteriori." EM is often presented (but not in this class!) as a maximum likelihood technique and it's easy to conflate the two. We've been discussing EM for MAP inference. In the context of the GMM:

  1. Maximum likelihood $\Rightarrow \arg\max \;\ln p(x|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})$
  2. Maximum a posteriori $\Rightarrow \arg\max \;\ln p(x, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = \arg\max \;\ln p(x|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) + \ln p(\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})$

- EM is simply the procedure of adding "hidden data" or extra latent variables to this such that the marginal is correct. That is, adding c such that $\int p(x, c|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) dc = p(x|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})$ or $\int p(x, c, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) dc = p(x, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})$ as the case may be, and then using these c to get closed form updates for $\pi, \boldsymbol{\mu}$, and $\boldsymbol{\Lambda}$. Therefore, people will sometimes say "maximum likelihood EM" or "MAP-EM" to be more specific. We'll just say "EM," but notice that up to now we've been doing MAP-EM, whereas today we will focus on ML-EM.

ML-EM for the GMM

- Since the GMM is always derived in the maximum likelihood framework for EM, we'll do the same here and not define prior distributions on π, $\boldsymbol{\mu}$ or $\boldsymbol{\Lambda}$. We'll then discuss variational inference where we will need to introduce priors.

- The log likelihood $\ln p(x|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})$ is a problem because the sum-log-sum form does not lead to closed form updates of parameters. To avoid gradient methods, we use EM, which means we must introduce an additional variable to the model. We start from the end and define a larger model, then show that it has the correct marginal.

- Model (expanded): For each $x_i \in \mathbb{R}^d$, given π, $\boldsymbol{\mu}$ and $\boldsymbol{\Lambda}$, generate

$$c_i \sim \text{Discrete}(\pi), \qquad x_i \,|\, c_i \sim \text{Normal}(\mu_{c_i}, \Lambda_{c_i}^{-1}) \tag{3}$$

- Notice that this is similar to LDA, where we used an indicator to say which topic a word came from. The function of this indicator is identical here. The value of $c_i \in \{1, \dots, K\}$ says which of the K Gaussians generated x_i, as is evident because c_i picks out the parameters of the corresponding Gaussian.

Marginal distribution

- We now need to check that this expanded model has the correct marginal distribution. We have to do this because we set out to maximize

$$\prod_{i=1}^{n} p(x_i|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = \prod_{i=1}^{n}\sum_{j=1}^{n} \pi_j \text{Normal}(x_i|\mu_j, \Lambda_j^{-1}) \tag{4}$$

- The new extended model has the likelihood

$$\begin{aligned}\prod_{i=1}^{n} p(x_i, c_i|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) &= \prod_{i=1}^{n} p(x_i|c_i, \boldsymbol{\mu}, \boldsymbol{\Lambda}) p(c_i|\pi) \\ &= \prod_{i=1}^{n}\prod_{j=1}^{K} \left(\pi_j \text{Normal}(x_i|\mu_j, \Lambda_j^{-1})\right)^{\mathbb{1}(c_i=j)} \end{aligned}\tag{5}$$

- Notice that indicators are used to pick out the correct term like in LDA. The next question is, what is

$$p(x|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = \sum_{c_1=1}^{K}\cdots\sum_{c_n=1}^{K}\prod_{i=1}^{n} p(x_i, c_i|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = \prod_{i=1}^{n}\sum_{j=1}^{K} p(x_i, c_i = j|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) \tag{6}$$

- It might seem trivial, but we literally just plug in and sum

$$\prod_{i=1}^{n}\sum_{j=1}^{K} \underbrace{p(x_i|c_i = j, \boldsymbol{\mu}, \boldsymbol{\Lambda})}_{=\,\text{Normal}(\mu_j, \Lambda_j^{-1})} \underbrace{p(c_i = j|\pi)}_{=\,\pi_j} \tag{7}$$

- This is exactly the marginal we want, so we found our extra variable.

Deriving an EM algorithm for the GMM

- We start where we have to start, with the EM equation

$$\ln p(x|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = \sum_{\boldsymbol{c}} q(\boldsymbol{c}) \ln \frac{p(x, \boldsymbol{c}|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})}{q(\boldsymbol{c})} + \sum_{\boldsymbol{c}} q(\boldsymbol{c}) \ln \frac{q(\boldsymbol{c})}{p(\boldsymbol{c}|x, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})} \tag{8}$$

- We make a few comments about this equality:

  1. The extra variables $\boldsymbol{c} = (c_1, \ldots, c_n)$ are discrete. Therefore there is no integral because c isn't in a continuous space. Sums are used in discrete settings and integrals in continuous settings, but the path to the equality is identical. In our previous derivation of this equation, we can literally replace every $\int$ with a Σ and remove the infinitesimals (e.g., dc)

2. The sum is over all possible values of the vector $\boldsymbol{c}$, where each entry $c_i \in \{1, \ldots, K\}$. Therefore, as written, there are K^n vectors we have to sum over. Of course, this is impossible for almost any problem (K^n is massive!) so if we can't simplify the problem, then EM is useless.

3. Despite the combinatorial issue, this is the true, correct equation. We have to start from here and try to simplify. Unlike variational methods where we make approximations with respect to $q(\boldsymbol{c})$—i.e., we define $q(\boldsymbol{c}) = \prod_i q(c_i)$—in EM we must set $q(\boldsymbol{c}) = p(\boldsymbol{c}|x, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})$ and so the true conditional posterior on all c_i dictates what we can do about $q(\boldsymbol{c})$.

4. Next we will show that the true conditional posterior factorizes, and so therefore $q(\boldsymbol{c})$ must factorize as well (i.e., there's no approximation in factorizing $q(\boldsymbol{c})$ as above). The posterior tells us that this is the case, and so no approximations are being made w.r.t. $q(\boldsymbol{c})$.

5. Also, as a side comment, notice that even when doing maximum likelihood where no priors are assumed on the model, posterior calculation using Bayes rule still factors into the problem. Therefore, even though we're not doing Bayesian modeling here, Bayesian ideas are still critical for making progress on this problem because we want to use EM and EM needs Bayes rule.

- We recall the steps of the EM algorithm and then work through each step.

  E-Step

  1. Set $q(\boldsymbol{c}) = p(\boldsymbol{c}|x, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})$
  2. Calculate $\sum_{\boldsymbol{c}} q(\boldsymbol{c}) \ln p(x, \boldsymbol{c}|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})$

  M-Step

  3. Maximize #2 over π, $\boldsymbol{\mu}$ and $\boldsymbol{\Lambda}$

- E-Step (Step 1): Use Bayes rule to calculate the conditional posterior distribution

$$\begin{aligned} p(\boldsymbol{c}|x, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) &\propto p(x|\boldsymbol{c}, \boldsymbol{\mu}, \boldsymbol{\Lambda})p(\boldsymbol{c}|\pi) \\ &\propto \prod_{i=1}^{n} p(x_i|c_i, \boldsymbol{\mu}, \boldsymbol{\Lambda})p(c_i|\pi) \end{aligned} \tag{9}$$

- This conditional independence removes the K^n combinatorial issue. We can divide this by a number $Z = \prod_{i=1}^n Z_i$ to normalize the entire function of $\boldsymbol{c}$ by normalizing each individual prior-likelihood pair,

$$\begin{aligned} p(\boldsymbol{c}|x, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) &= \frac{1}{Z}\prod_{i=1}^{n} p(x_i|c_i, \boldsymbol{\mu}, \boldsymbol{\Lambda})p(c_i|\pi) \\ &= \prod_{i=1}^{n} \frac{p(x_i|c_i, \boldsymbol{\mu}, \boldsymbol{\Lambda})p(c_i|\pi)}{Z_i} \\ &= \prod_{i=1}^{n} p(c_i|x_i, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) \end{aligned} \tag{10}$$

where

$$\begin{aligned} p(c_i = k|x_i, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) &= \frac{p(x_i|c_i = k, \boldsymbol{\mu}, \boldsymbol{\Lambda})p(c_i = k|\pi)}{\sum_{j=1}^K p(x_i|c_i = j, \boldsymbol{\mu}, \boldsymbol{\Lambda})p(c_i = j|\pi)} \\ &= \frac{\pi_k \text{Normal}(x_i|\mu_k, \Lambda_k^{-1})}{\sum_{j=1}^K \pi_j \text{Normal}(x_i|\mu_j, \Lambda_j^{-1})} \end{aligned} \tag{11}$$

- So to summarize

$$q(\boldsymbol{c}) \;=\; p(\boldsymbol{c}|x, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) \;=\; \prod_{i=1}^n \underbrace{p(c_i|x_i, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})}_{\to q(c_i)} \;=\; \prod_{i=1}^n q(c_i) \tag{12}$$

- We use notation $q(c_i = j) = \phi_i(j)$. When implementing, we first set $\phi_i(j) \leftarrow \pi_j \text{Normal}(x_i|\mu_j, \Lambda_j^{-1})$ for $j = 1, \dots, K$ and then normalize the K-dimensional vector ϕ_i by dividing it by its sum.

- <u>E-Step (Step 2)</u>: The expectation we need to take is

$$\mathcal{L}(\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = \sum_{i=1}^n \mathbb{E}_{q(\boldsymbol{c})}[\ln p(x_i, c_i|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})] + \text{const.} \tag{13}$$

  The constant is with respect to $\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}$.

- Because each (x_i, c_i) are conditionally independent of each other. The expectation over $q(\boldsymbol{c})$ reduces to an expectation over $q(c_i)$. It's worth thinking about this. Each of the n expectations above is over the q distribution on the entire vector $\boldsymbol{c} = (c_1, \dots, c_n)$. However, because the ith term only contains c_i in it, the sum over $c_{i'}$ $i' \neq i$ in $q(\boldsymbol{c})$ causes each $q(c_{i'})$ to disappear because it sums to one. We're then left with:

$$\mathcal{L}(\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = \sum_{i=1}^n \mathbb{E}_{q(c_i)}[\ln p(x_i, c_i|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})] + \text{const.} \tag{14}$$

- Notice that this takes care of the combinatorial problem. We now only need to sum over $K \cdot n$ terms instead of over K^n terms.

- Continuing,

$$\begin{aligned} \mathcal{L}(\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) &= \sum_{i=1}^n \sum_{j=1}^K q(c_i = j)[\ln p(x_i|c_i = j, \boldsymbol{\mu}, \boldsymbol{\Lambda}) + \ln p(c_i = j|\pi)] + \text{const.} \qquad (15) \\ &= \sum_{i=1}^n \sum_{j=1}^K \phi_i(j) \left(\frac{1}{2} \ln |\Lambda_j| - \frac{1}{2}(x_i - \mu_j)^T \Lambda_j (x_i - \mu_j) + \ln \pi_j \right) + \text{const.} \end{aligned}$$

- (The constant in the second line contains extra terms in addition to what's in the first line.)

- M-Step (Step 3): Finally, we maximize $\mathcal{L}(\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})$ over its parameters. We hope we can do this by taking derivatives and setting to zero. If not, then EM probably hasn't helped us.

  a) $\nabla_{\mu_j}\mathcal{L} = 0$

$$\nabla_{\mu_j}\mathcal{L} \;=\; -\sum_{i=1}^{n}\phi_i(j)(\Lambda_j\mu_j - \Lambda_j x_j) = 0 \tag{16}$$

$$\Downarrow$$

$$\mu_j = \frac{1}{n_j}\sum_{i=1}^{n}\phi_i(j)x_i, \qquad n_j = \sum_{i=1}^{n}\phi_i(j) \tag{17}$$

  Notice that this is a weighted average of all the points in the data set. The weight is determined by the conditional posterior probability of each point coming from the jth cluster using the parameters from the previous iteration.

  b) $\nabla_{\Lambda_j}\mathcal{L} = 0$

$$\nabla_{\Lambda_j}\mathcal{L} \;=\; \sum_{i=1}^{n}\phi_i(j)(\frac{1}{2}\Lambda_j^{-1} - \frac{1}{2}(x_i-\mu_j)(x_i-\mu_j)^T) = 0 \tag{18}$$

$$\Downarrow$$

$$\Lambda_j^{-1} = \frac{1}{n_j}\sum_{i=1}^{n}\phi_i(j)(x_i-\mu_j)(x_i-\mu_j)^T, \qquad n_j = \sum_{i=1}^{n}\phi_i(j) \tag{19}$$

  We use the equalities $\nabla_\Lambda \ln|\Lambda| = \Lambda^{-1}$ and $x^T\Lambda x = \text{trace}(\Lambda x x^T) \Rightarrow \nabla_\Lambda x^T\Lambda x = xx^T$. Just like the mean μ_j we see the inverse of Λ_j (i.e., the covariance) is equal to a weighted version of the empirical covariance. For this step, we use the most recent update for μ_j.

  c) $\nabla_\pi\mathcal{L} = 0$ subject to $\pi_j \geq 0$ and $\sum_{j=1}^{K}\pi_j = 1$. This step requires Lagrange multipliers to ensure these constraints. We won't review the technique of Lagrange multipliers since it would be too much of a digression, but it's important to know in general. The solution is

$$\pi_j = \frac{n_j}{n}, \qquad n_j = \sum_{i=1}^{n}\phi_i(j) \tag{20}$$

  The update of the prior probability that a point comes from cluster j is the fraction of points we expect to see from cluster j under the current posterior distribution.

- Notice we don't need to iterate over $\boldsymbol{\mu}$, $\boldsymbol{\Lambda}$, π to maximize $\mathcal{L}$.

- We are left with the following EM algorithm.

A maximum likelihood EM algorithm for the Gaussian mixture model

Input: Data $x_1, \ldots, x_n$, $x \in \mathbb{R}^d$. Number of clusters K.

Output: GMM parameters π, $\boldsymbol{\mu}$, $\boldsymbol{\Lambda}$ and cluster assignment distributions ϕ_i

1. Initialize $\pi^{(0)}$ and each $(\mu_j^{(0)}, \Lambda_j^{(0)})$ in some way.
2. At iteration t,
   (a) E-Step: For $i = 1, \ldots, n$ and $j = 1, \ldots, K$ set

$$\phi_i^{(t)}(j) = \frac{\pi_j^{(t-1)}\text{Normal}(x_i|\mu_j^{(t-1)}, (\Lambda_j^{(t-1)})^{-1})}{\sum_{k=1}^K \pi_k^{(t-1)}\text{Normal}(x_i|\mu_k^{(t-1)}, (\Lambda_k^{(t-1)})^{-1})}$$

3. M-Step: Set

$$\begin{aligned}
n_j^{(t)} &= \sum_{i=1}^n \phi_i^{(t)}(j) \\
\mu_j^{(t)} &= \frac{1}{n_j^{(t)}} \sum_{i=1}^n \phi_i^{(t)}(j) x_i \\
\Lambda_j^{(t)} &= \left(\frac{1}{n_j^{(t)}} \sum_{i=1}^n \phi_i^{(t)}(j)(x_i - \mu_j^{(t)})(x_i - \mu_j^{(t)})^T \right)^{-1} \\
\pi_j^{(t)} &= \frac{n_j^{(t)}}{n}
\end{aligned}$$

4. Calculate $f_t = \ln p(x|\pi^{(t)}, \boldsymbol{\mu}^{(t)}, \boldsymbol{\Lambda}^{(t)})$ to assess convergence of f as a function of t.

Variational inference (VI)

- We next look at VI for the GMM. We therefore need to define priors for π, $\boldsymbol{\mu}$ and $\boldsymbol{\Lambda}$.
- Priors: We have many options for priors, but we choose (out of convenience)

$$\pi \sim \text{Dirichlet}(\alpha), \quad \mu_j \sim \text{Normal}(0, cI), \quad \Lambda_j \sim \text{Wishart}(a, B) \tag{21}$$

- The Wishart distribution
- Since there's a good chance the Wishart distribution is new, we'll review it briefly here. The Wishart distribution is a probability distribution on positive definite matrices. It is the conjugate prior for the precision, Λ_j, of a multivariate Gaussian, which is why we pick it. For a $d \times d$ matrix Λ, the density function is

$$p(\Lambda|a, B) = \frac{|\Lambda|^{\frac{a-d-1}{2}} e^{-\frac{1}{2}\text{trace}(B\Lambda)}}{2^{\frac{ad}{2}}|B|^{-\frac{a}{2}}\Gamma_d(a/2)} \tag{22}$$

The parameter requirements are $a > d-1$ and B is a positive definite $d \times d$ matrix. The function $\Gamma_d(a/2)$ is a complicated function of the gamma function,

$$\Gamma_d(a/2) = \pi^{\frac{d(d-1)}{4}} \prod_{j=1}^{d} \Gamma\left(\frac{a}{2} + \frac{1-j}{2}\right)$$

- The important expectations for VI are of sufficient statistics of the Wishart distribution. Since the Wishart distribution is an exponential family distribution, we can use the technique discussed in a previous lecture to find that

$$\mathbb{E}[\Lambda] = aB^{-1}, \quad \mathbb{E}[\ln|\Lambda|] = d\ln 2 - \ln|B| + \sum_{j=1}^{d} \psi(a/2 + (1-j)/2) \tag{23}$$

$\psi(\cdot)$ is the digamma function discussed previously. It's something you would call a built-in function to evaluate in your code.

- It's worth trying the previously discussed exponential family trick out for $\mathbb{E}[\ln|\Lambda|]$ to see how easily we can derive this nasty expectation.
- Joint and log joint likelihood: The joint likelihood and log joint likelihood can be written as

$$p(x, \boldsymbol{c}, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = \Big[\prod_{i=1}^{n}\prod_{j=1}^{K} p(x_i|c_i = j, \boldsymbol{\mu}, \boldsymbol{\Lambda})p(c_i = j|\pi)\Big]\Big[\prod_{j=1}^{K} p(\mu_j)p(\Lambda_j)\Big]p(\pi) \tag{24}$$

Therefore, using the indicator function representation for c_i,

$$\ln p(x, \boldsymbol{c}, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = \sum_{i=1}^{n}\sum_{j=1}^{K} \mathbb{1}(c_i = j)[\ln p(x_i|\mu_j, \Lambda_j) + \ln \pi_j] + \sum_{j=1}^{K}[\ln p(\mu_j) + \ln p(\Lambda_j)] + \ln p(\pi) \tag{25}$$

- Posterior calculation: Since we can't calculate $p(\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{c}|x)$ we use a variational approximation with mean-field factorization

$$p(\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{c}|x) \;\approx\; q(\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{c}) \;=\; q(\pi)\left[\prod_{j=1}^{K} q(\mu_j)q(\Lambda_j)\right]\left[\prod_{i=1}^{n} q(c_i)\right] \tag{26}$$

- Next we need to define the distribution family of each q. We know how to do this, but notice that we can make a CEF argument. All distributions are exponential family distributions (Gaussian, Wishart, Dirichlet, multinomial) and we can verify that the conditional posterior of each model variable is in the same family as the prior, hence there is conditional conjugacy. The model is a conjugate exponential family model and so each q distribution should be set to the same family as the prior.

- We still need to calculate how to update their parameters, so the CEF argument hasn't really reduced the amount of work we have to do. However, we can say at the outset that

  - $q(\pi) =$ Dirichlet$(\alpha'_1, \ldots, \alpha'_K)$
  - $q(\mu_j) =$ Normal(m'_j, Σ'_j)
  - $q(\Lambda_j) =$ Wishart(a'_j, B'_j)
  - $q(c_i) =$ Multinomial(ϕ_i)

- The variational inference algorithm will then consist of iterating through each of these q distributions and modifying their parameter values to make their product closer to the true posterior distribution according to their KL divergence.

- Since this is a CEF model, we can use the "log-expectation-exponentiate-normalize" approach we've been discussing over the past several lectures.

- $q(c_i = j)$:

$$\begin{aligned} q(c_i = j) &\propto e^{\mathbb{E}[\ln p(x_i, c_i = j|\pi, \mu_j, \Lambda_j)]} \\ &\propto e^{\mathbb{E}[\ln p(x_i|\mu_j, \Lambda_j)] + \mathbb{E}[\ln p(c_i = j|\pi)]} \\ &\propto e^{\frac{1}{2}\mathbb{E}[\ln|\Lambda_j|] - \frac{1}{2}\mathbb{E}[(x_i-\mu_j)^T\Lambda_j(x_i-\mu_j)] + \mathbb{E}[\ln \pi_j]} \end{aligned} \tag{27}$$

  The expectation is over π, μ_j and Λ_j. They are taken using the q distributions on them, so

$$\mathbb{E}[\ln \pi_j] = \psi(\alpha'_j) - \psi(\textstyle\sum_k \alpha'_k)$$

  Multiplying out we see that

$$\mathbb{E}[(x_i - \mu_j)^T\Lambda_j(x_i - \mu_j)] = (x_i - \mathbb{E}[\mu_j])^T\mathbb{E}[\Lambda_j](x_i - \mathbb{E}[\mu_j]) + \text{trace}(\mathbb{E}[\Lambda_j]\Sigma'_j)$$

  We discussed $\mathbb{E}[\ln|\Lambda_j|]$ above. Σ'_j is from $q(\mu_j)$ and $\mathbb{E}[\mu_j] = m'_j$.

- $\underline{q(\pi)}$:

$$q(\pi) \propto e^{\sum_{i=1}^n \mathbb{E}[\ln p(c_j|\pi)] + \ln p(\pi)}$$
$$\propto e^{\sum_{i=1}^n \sum_{j=1}^K \phi_i(j) \ln \pi_j + \sum_{i=1}^K (\alpha-1)\ln \pi_j}$$
$$\propto \prod_{j=1}^K \pi_j^{\alpha + \sum_{i=1}^n \phi_i(j) - 1} \tag{28}$$

So we set

$$\alpha'_j = \alpha + n_j, \qquad n_j = \sum_{i=1}^n \phi_i(j)$$

- $\underline{q(\mu_j)}$:

$$q(\mu_j) \propto e^{\sum_{i=1}^n \mathbb{E}[\ln p(x_i|c_i=j,\mu_j,\Lambda_j)] + \ln p(\mu_j)}$$
$$\propto e^{-\frac{1}{2}\sum_{i=1}^n \phi_i(j)(x_i-\mu_j)^T \mathbb{E}[\Lambda_j](x_i-\mu_j) - \frac{1}{2c}\mu_j^T\mu_j}$$
$$\propto e^{-\frac{1}{2}\sum_{i=1}^n (x_i-\mu_j)^T (\phi_i(j)\mathbb{E}[\Lambda_j])(x_i-\mu_j) - \frac{1}{2c}\mu_j^T\mu_j} \tag{29}$$

Normalizing this over μ, the result is a Gaussian distribution $q(\mu_j) = \text{Normal}(m'_j, \Sigma'_j)$ where

$$\Sigma'_j = \left(c^{-1}I + n_j\mathbb{E}[\Lambda_j]\right)^{-1}, \qquad m'_j = \Sigma'_j\left(\mathbb{E}[\Lambda_j]\sum_{i=1}^n \phi_i(j)x_i\right), \qquad n_j = \sum_{i=1}^n \phi_i(j) \tag{30}$$

- $\underline{q(\Lambda_j)}$:

$$q(\Lambda_j) \propto e^{\sum_{i=1}^n \mathbb{E}[\ln p(x_i|c_i=j,\mu_j,\Lambda_j)] + \ln p(\Lambda_j)}$$
$$\propto |\Lambda_j|^{\frac{n_j+a-d-1}{2}} e^{-\frac{1}{2}\sum_{i=1}^n \phi_i(j)\left[(x_i-\mathbb{E}[\mu_j])^T\Lambda_j(x_i-\mathbb{E}[\mu_j]) + \text{trace}(\Lambda_j\Sigma'_j)\right] - \frac{1}{2}\text{trace}(B\Lambda_j)} \tag{31}$$

We can see this has a Wishart form, $q(\Lambda_j) = \text{Wishart}(a'_j, B'_j)$ where

$$a'_j = a + n_j, \qquad B'_j = B + \sum_{i=1}^n \phi_i(j)\left[(x_i + \mathbb{E}[\mu_j])(x_i - \mathbb{E}[\mu_j])^T + \Sigma'_j\right] \tag{32}$$

And again $n_j = \sum_{i=1}^n \phi_i(j)$. We used the fact that

$$\phi_i(j)(x_i - \mathbb{E}[\mu_j])^T\Lambda_j(x_i - \mathbb{E}[\mu_j]) = \text{tr}(\phi_i(j)(x_i - \mathbb{E}[\mu_j])(x_i - \mathbb{E}[\mu_j])^T\Lambda_j)$$

- In all of these above equations, the expectations are taken using the q distributions, so for example $\mathbb{E}[\mu_j] = m'_j$ using the most recent value of m'_j.

- The result is the following algorithm. I remove the $'$ below because the notation already looks very complicated. I give iteration indexes to these parameters to emphasize the iterative dependence, which makes it look complicated.

A variational inference algorithm for the Gaussian mixture model

Input: Data $x_1, \dots, x_n$, $x \in \mathbb{R}^d$. Number of clusters K.

Output: Parameters for $q(\pi)$, $q(\mu_j)$, $q(\Lambda_j)$ and $q(c_i)$.

1. Initialize $(\alpha_1^{(0)}, \dots, \alpha_K^{(0)})$, $(m_j^{(0)}, \Sigma_j^{(0)})$, $(a_j^{(0)}, B_j^{(0)})$ in some way.
2. At iteration t,

   (a) Update $q(c_i)$ for $i = 1, \dots, n$. This one is complicated so we break it down. Set

   $$\phi_i^{(t)}(j) = \frac{\mathrm{e}^{\frac{1}{2}t_1(j) - \frac{1}{2}t_2(j) - \frac{1}{2}t_3(j) + t_4(j)}}{\sum_{k=1}^K \mathrm{e}^{\frac{1}{2}t_1(k) - \frac{1}{2}t_2(k) - \frac{1}{2}t_3(k) + t_4(k)}}$$

   where

   $$t_1(j) = \sum_{k=1}^d \psi\left(\frac{1 - k + a_j^{(t-1)}}{2}\right) - \ln|B_j^{(t-1)}|$$

   $$t_2(j) = (x_i - m_j^{(t-1)})^T (a_j^{(t-1)} (B_j^{(t-1)})^{-1})(x_i - m_j^{(t-1)})$$

   $$t_3(j) = \text{trace}(a_j^{(t-1)} (B_j^{(t-1)})^{-1} \Sigma_j^{(t-1)}), \qquad t_4(j) = \psi(\alpha_j^{(t-1)}) - \psi(\sum_k \alpha_k^{(t-1)})$$

   Tip: Some of these values can be calculated once and reused for each $q(c_i)$.

   (b) Set $n_j^{(t)} = \sum_{i=1}^n \phi_i^{(t)}(j)$ for $j = 1, \dots, K$

   (c) Update $q(\pi)$ by setting

   $$\alpha_j^{(t)} = \alpha + n_j^{(t)}$$

   for $j = 1, \dots, K$.

   (d) Update $q(\mu_j)$ for $j = 1, \dots, K$ by setting

   $$\Sigma_j^{(t)} = \left(c^{-1}I + n_j^{(t)} a_j^{(t-1)} (B_j^{(t-1)})^{-1}\right)^{-1}, \qquad m_j^{(t)} = \Sigma_j^{(t)} \left(a_j^{(t-1)} (B_j^{(t-1)})^{-1} \sum_{i=1}^n \phi_i^{(t)}(j) x_i\right)$$

   (e) Update $q(\Lambda_j)$ for $j = 1, \dots, K$ by setting

   $$a_j^{(t)} = a + n_j^{(t)}, \qquad B_j^{(t)} = B + \sum_{i=1}^n \phi_i^{(t)}(j) \left[(x_i - m_j^{(t)})(x_i - m_j^{(t)})^T + \Sigma_j^{(t)}\right]$$

   (f) Calculate the variational objective function to assess convergence as a function of iteration:

   $$\mathcal{L} = \mathbb{E}[\ln p(x, c, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})] - \mathbb{E}[\ln q]$$