Lecture notes

Part IX
The EM algorithm

In the previous set of notes, we talked about the EM algorithm as applied to fitting a mixture of Gaussians. In this set of notes, we give a broader view of the EM algorithm, and show how it can be applied to a large family of estimation problems with latent variables. We begin our discussion with a very useful result called **Jensen's inequality**

1 Jensen's inequality

Let f be a function whose domain is the set of real numbers. Recall that f is a convex function if $f''(x) \geq 0$ (for all $x \in \mathbb{R}$). In the case of f taking vector-valued inputs, this is generalized to the condition that its hessian H is positive semi-definite ($H \geq 0$). If $f''(x) > 0$ for all x, then we say f is **strictly** convex (in the vector-valued case, the corresponding statement is that H must be positive definite, written $H > 0$). Jensen's inequality can then be stated as follows:

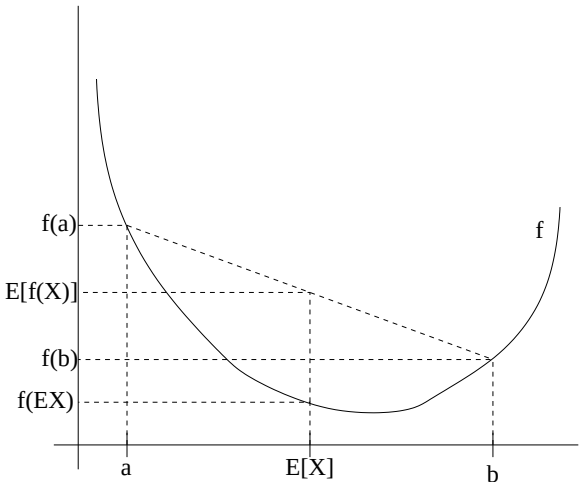
Theorem. Let f be a convex function, and let X be a random variable. Then:

$$\mathrm{E}[f(X)] \geq f(\mathrm{E}X).$$

Moreover, if f is strictly convex, then $\mathrm{E}[f(X)] = f(\mathrm{E}X)$ holds true if and only if $X = \mathrm{E}[X]$ with probability 1 (i.e., if X is a constant).

Recall our convention of occasionally dropping the parentheses when writing expectations, so in the theorem above, $f(\mathrm{E}X) = f(\mathrm{E}[X])$.

For an interpretation of the theorem, consider the figure below.



Here, f is a convex function shown by the solid line. Also, X is a random variable that has a 0.5 chance of taking the value a, and a 0.5 chance of taking the value b (indicated on the x-axis). Thus, the expected value of X is given by the midpoint between a and b.

We also see the values $f(a)$, $f(b)$ and $f(\mathrm{E}[X])$ indicated on the y-axis. Moreover, the value $\mathrm{E}[f(X)]$ is now the midpoint on the y-axis between $f(a)$ and $f(b)$. From our example, we see that because f is convex, it must be the case that $\mathrm{E}[f(X)] \geq f(\mathrm{E}X)$.

Incidentally, quite a lot of people have trouble remembering which way the inequality goes, and remembering a picture like this is a good way to quickly figure out the answer.

Remark. Recall that f is [strictly] concave if and only if $-f$ is [strictly] convex (i.e., $f''(x) \leq 0$ or $H \leq 0$). Jensen's inequality also holds for concave functions f, but with the direction of all the inequalities reversed ($\mathrm{E}[f(X)] \leq f(\mathrm{E}X)$, etc.).

2 The EM algorithm

Suppose we have an estimation problem in which we have a training set $\{x^{(1)}, \ldots, x^{(n)}\}$ consisting of n independent examples. We have a latent variable model $p(x, z; \theta)$ with z being the latent variable (which for simplicity is assumed to take finite number of values). The density for x can be obtained by marginalized over the latent variable z:

$$p(x; \theta) = \sum_z p(x, z; \theta) \tag{1}$$

We wish to fit the parameters θ by maximizing the log-likelihood of the data, defined by

$$\ell(\theta) = \sum_{i=1}^{n} \log p(x^{(i)}; \theta) \tag{2}$$

We can rewrite the objective in terms of the joint density $p(x, z; \theta)$ by

$$\ell(\theta) = \sum_{i=1}^{n} \log p(x^{(i)}; \theta) \tag{3}$$

$$= \sum_{i=1}^{n} \log \sum_{z^{(i)}} p(x^{(i)}, z^{(i)}; \theta). \tag{4}$$

But, explicitly finding the maximum likelihood estimates of the parameters θ may be hard since it will result in difficult non-convex optimization problems.[1] Here, the $z^{(i)}$'s are the latent random variables; and it is often the case that if the $z^{(i)}$'s were observed, then maximum likelihood estimation would be easy.

In such a setting, the EM algorithm gives an efficient method for maximum likelihood estimation. Maximizing $\ell(\theta)$ explicitly might be difficult, and our strategy will be to instead repeatedly construct a lower-bound on ℓ (E-step), and then optimize that lower-bound (M-step).[2]

It turns out that the summation $\sum_{i=1}^{n}$ is not essential here, and towards a simpler exposition of the EM algorithm, we will first consider optimizing the the likelihood $\log p(x)$ for **a single example** x. After we derive the algorithm for optimizing $\log p(x)$, we will convert it to an algorithm that works for n examples by adding back the sum to each of the relevant equations. Thus, now we aim to optimize $\log p(x; \theta)$ which can be rewritten as

$$\log p(x; \theta) = \log \sum_{z} p(x, z; \theta) \tag{5}$$

[1] It's mostly an empirical observation that the optimization problem is difficult to optimize.

[2] Empirically, the E-step and M-step can often be computed more efficiently than optimizing the function $\ell(\cdot)$ directly. However, it doesn't necessarily mean that alternating the two steps can always converge to the global optimum of $\ell(\cdot)$. Even for mixture of Gaussians, the EM algorithm can either converge to a global optimum or get stuck, depending on the properties of the training data. Empirically, for real-world data, often EM can converge to a solution with relatively high likelihood (if not the optimum), and the theory behind it is still largely not understood.

Let Q be a distribution over the possible values of z. That is, $\sum_z Q(z) = 1$, $Q(z) \geq 0$).

Consider the following:[3]

$$\begin{aligned} \log p(x;\theta) &= \log \sum_z p(x,z;\theta) \\ &= \log \sum_z Q(z) \frac{p(x,z;\theta)}{Q(z)} \qquad (6) \\ &\geq \sum_z Q(z) \log \frac{p(x,z;\theta)}{Q(z)} \qquad (7) \end{aligned}$$

The last step of this derivation used Jensen's inequality. Specifically, $f(x) = \log x$ is a concave function, since $f''(x) = -1/x^2 < 0$ over its domain $x \in \mathbb{R}^+$. Also, the term

$$\sum_z Q(z) \left[\frac{p(x,z;\theta)}{Q(z)} \right]$$

in the summation is just an expectation of the quantity $[p(x,z;\theta)/Q(z)]$ with respect to z drawn according to the distribution given by Q.[4] By Jensen's inequality, we have

$$f\left(\mathrm{E}_{z\sim Q} \left[\frac{p(x,z;\theta)}{Q(z)} \right] \right) \geq \mathrm{E}_{z\sim Q} \left[f\left(\frac{p(x,z;\theta)}{Q(z)} \right) \right],$$

where the "$z \sim Q$" subscripts above indicate that the expectations are with respect to z drawn from Q. This allowed us to go from Equation (6) to Equation (7).

Now, for **any** distribution Q, the formula (7) gives a lower-bound on $\log p(x;\theta)$. There are many possible choices for the Q's. Which should we choose? Well, if we have some current guess θ of the parameters, it seems natural to try to make the lower-bound tight at that value of θ. I.e., we will make the inequality above hold with equality at our particular value of θ.

To make the bound tight for a particular value of θ, we need for the step involving Jensen's inequality in our derivation above to hold with equality.

[3] If z were continuous, then Q would be a density, and the summations over z in our discussion are replaced with integrals over z.

[4] We note that the notion $\frac{p(x,z;\theta)}{Q(z)}$ only makes sense if $Q(z) \neq 0$ whenever $p(x,z;\theta) \neq 0$. Here we implicitly assume that we only consider those Q with such a property.

For this to be true, we know it is sufficient that the expectation be taken over a "constant"-valued random variable. I.e., we require that

$$\frac{p(x, z; \theta)}{Q(z)} = c$$

for some constant c that does not depend on z. This is easily accomplished by choosing

$$Q(z) \propto p(x, z; \theta).$$

Actually, since we know $\sum_z Q(z) = 1$ (because it is a distribution), this further tells us that

$$\begin{aligned} Q(z) &= \frac{p(x, z; \theta)}{\sum_z p(x, z; \theta)} \\ &= \frac{p(x, z; \theta)}{p(x; \theta)} \\ &= p(z|x; \theta) \end{aligned} \tag{8}$$

Thus, we simply set the Q's to be the posterior distribution of the z's given x and the setting of the parameters θ.

Indeed, we can directly verify that when $Q(z) = p(z|x; \theta)$, then equation (7) is an equality because

$$\begin{aligned} \sum_z Q(z) \log \frac{p(x, z; \theta)}{Q(z)} &= \sum_z p(z|x; \theta) \log \frac{p(x, z; \theta)}{p(z|x; \theta)} \\ &= \sum_z p(z|x; \theta) \log \frac{p(z|x; \theta)p(x; \theta)}{p(z|x; \theta)} \\ &= \sum_z p(z|x; \theta) \log p(x; \theta) \\ &= \log p(x; \theta) \sum_z p(z|x; \theta) \\ &= \log p(x; \theta) \qquad \text{(because } \textstyle\sum_z p(z|x; \theta) = 1\text{)} \end{aligned}$$

For convenience, we call the expression in Equation (7) the **evidence lower bound** (ELBO) and we denote it by

$$\text{ELBO}(x; Q, \theta) = \sum_z Q(z) \log \frac{p(x, z; \theta)}{Q(z)} \tag{9}$$

With this equation, we can re-write equation (7) as

$$\forall Q, \theta, x, \quad \log p(x;\theta) \geq \text{ELBO}(x; Q, \theta) \tag{10}$$

Intuitively, the EM algorithm alternatively updates Q and θ by a) setting $Q(z) = p(z|x;\theta)$ following Equation (8) so that $\text{ELBO}(x; Q, \theta) = \log p(x;\theta)$ for x and the current θ, and b) maximizing $\text{ELBO}(x; Q, \theta)$ w.r.t θ while fixing the choice of Q.

Recall that all the discussion above was under the assumption that we aim to optimize the log-likelihood $\log p(x;\theta)$ for a single example x. It turns out that with multiple training examples, the basic idea is the same and we only needs to take a sum over examples at relevant places. Next, we will build the evidence lower bound for multiple training examples and make the EM algorithm formal.

Recall we have a training set $\{x^{(1)}, \ldots, x^{(n)}\}$. Note that the optimal choice of Q is $p(z|x;\theta)$, and it depends on the particular example x. Therefore here we will introduce n distributions $Q_1, \ldots, Q_n$, one for each example $x^{(i)}$. For each example $x^{(i)}$, we can build the evidence lower bound

$$\log p(x^{(i)};\theta) \geq \text{ELBO}(x^{(i)}; Q_i, \theta) = \sum_{z^{(i)}} Q_i(z^{(i)}) \log \frac{p(x^{(i)}, z^{(i)};\theta)}{Q_i(z^{(i)})}$$

Taking sum over all the examples, we obtain a lower bound for the log-likelihood

$$\begin{aligned} \ell(\theta) &\geq \sum_i \text{ELBO}(x^{(i)}; Q_i, \theta) \qquad (11) \\ &= \sum_i \sum_{z^{(i)}} Q_i(z^{(i)}) \log \frac{p(x^{(i)}, z^{(i)};\theta)}{Q_i(z^{(i)})} \end{aligned}$$

For *any* set of distributions $Q_1, \ldots, Q_n$, the formula (11) gives a lower-bound on $\ell(\theta)$, and analogous to the argument around equation (8), the Q_i that attains equality satisfies

$$Q_i(z^{(i)}) \quad = \quad p(z^{(i)}|x^{(i)};\theta)$$

Thus, we simply set the Q_i's to be the posterior distribution of the $z^{(i)}$'s given $x^{(i)}$ with the current setting of the parameters θ.

Now, for this choice of the Q_i's, Equation (11) gives a lower-bound on the loglikelihood ℓ that we're trying to maximize. This is the E-step. In the M-step of the algorithm, we then maximize our formula in Equation (11) with respect to the parameters to obtain a new setting of the θ's. Repeatedly carrying out these two steps gives us the EM algorithm, which is as follows:

Repeat until convergence {

(E-step) For each i, set

$$Q_i(z^{(i)}) := p(z^{(i)}|x^{(i)};\theta).$$

(M-step) Set

$$\begin{aligned}\theta &:= \arg\max_\theta \sum_{i=1}^{n} \mathrm{ELBO}(x^{(i)}; Q_i, \theta) \\ &= \arg\max_\theta \sum_i \sum_{z^{(i)}} Q_i(z^{(i)}) \log \frac{p(x^{(i)}, z^{(i)};\theta)}{Q_i(z^{(i)})}. \qquad (12)\end{aligned}$$

}

How do we know if this algorithm will converge? Well, suppose $\theta^{(t)}$ and $\theta^{(t+1)}$ are the parameters from two successive iterations of EM. We will now prove that $\ell(\theta^{(t)}) \leq \ell(\theta^{(t+1)})$, which shows EM always monotonically improves the log-likelihood. The key to showing this result lies in our choice of the Q_i's. Specifically, on the iteration of EM in which the parameters had started out as $\theta^{(t)}$, we would have chosen $Q_i^{(t)}(z^{(i)}) := p(z^{(i)}|x^{(i)};\theta^{(t)})$. We saw earlier that this choice ensures that Jensen's inequality, as applied to get Equation (11), holds with equality, and hence

$$\ell(\theta^{(t)}) = \sum_{i=1}^{n} \mathrm{ELBO}(x^{(i)}; Q_i^{(t)}, \theta^{(t)}) \qquad (13)$$

The parameters $\theta^{(t+1)}$ are then obtained by maximizing the right hand side of the equation above. Thus,

$$\begin{aligned}\ell(\theta^{(t+1)}) &\geq \sum_{i=1}^{n} \mathrm{ELBO}(x^{(i)}; Q_i^{(t)}, \theta^{(t+1)}) \\ &\qquad \text{(because ineqaulity (11) holds for all } Q \text{ and } \theta) \\ &\geq \sum_{i=1}^{n} \mathrm{ELBO}(x^{(i)}; Q_i^{(t)}, \theta^{(t)}) \qquad \text{(see reason below)} \\ &= \ell(\theta^{(t)}) \qquad \text{(by equation (13))}\end{aligned}$$

where the last inequality follows from that $\theta^{(t+1)}$ is chosen explicitly to be

$$\arg\max_\theta \quad \sum_{i=1}^{n} \mathrm{ELBO}(x^{(i)}; Q_i^{(t)}, \theta)$$

Hence, EM causes the likelihood to converge monotonically. In our description of the EM algorithm, we said we'd run it until convergence. Given the result that we just showed, one reasonable convergence test would be to check if the increase in $\ell(\theta)$ between successive iterations is smaller than some tolerance parameter, and to declare convergence if EM is improving $\ell(\theta)$ too slowly.

Remark. If we define (by overloading ELBO$(\cdot)$)

$$\text{ELBO}(Q, \theta) = \sum_{i=1}^{n} \text{ELBO}(x^{(i)}; Q_i, \theta) = \sum_i \sum_{z^{(i)}} Q_i(z^{(i)}) \log \frac{p(x^{(i)}, z^{(i)}; \theta)}{Q_i(z^{(i)})} \tag{14}$$

then we know $\ell(\theta) \geq \text{ELBO}(Q, \theta)$ from our previous derivation. The EM can also be viewed an alternating maximization algorithm on ELBO(Q, θ), in which the E-step maximizes it with respect to Q (check this yourself), and the M-step maximizes it with respect to θ.

2.1 Other interpretation of ELBO

Let $\text{ELBO}(x; Q, \theta) = \sum_z Q(z) \log \frac{p(x,z;\theta)}{Q(z)}$ be defined as in equation (9). There are several other forms of ELBO. First, we can rewrite

$$\begin{aligned} \text{ELBO}(x; Q, \theta) &= \text{E}_{z\sim Q}[\log p(x, z; \theta)] - \text{E}_{z\sim Q}[\log Q(z)] \\ &= \text{E}_{z\sim Q}[\log p(x|z; \theta)] - D_{KL}(Q \| p_z) \end{aligned} \tag{15}$$

where we use p_z to denote the marginal distribution of z (under the distribution $p(x, z; \theta)$), and $D_{KL}()$ denotes the KL divergence

$$D_{KL}(Q \| p_z) = \sum_z Q(z) \log \frac{Q(z)}{p(z)} \tag{16}$$

In many cases, the marginal distribution of z does not depend on the parameter θ. In this case, we can see that maximizing ELBO over θ is equivalent to maximizing the first term in (15). This corresponds to maximizing the conditional likelihood of x conditioned on z, which is often a simpler question than the original question.

Another form of ELBO$(\cdot)$ is (please verify yourself)

$$\text{ELBO}(x; Q, \theta) = \log p(x) - D_{KL}(Q \| p_{z|x}) \tag{17}$$

where $p_{z|x}$ is the conditional distribution of z given x under the parameter θ. This forms shows that the maximizer of ELBO(Q, θ) over Q is obtained when $Q = p_{z|x}$, which was shown in equation (8) before.

3 Mixture of Gaussians revisited

Armed with our general definition of the EM algorithm, let's go back to our old example of fitting the parameters ϕ, μ and Σ in a mixture of Gaussians. For the sake of brevity, we carry out the derivations for the M-step updates only for ϕ and μ_j, and leave the updates for Σ_j as an exercise for the reader.

The E-step is easy. Following our algorithm derivation above, we simply calculate

$$w_j^{(i)} = Q_i(z^{(i)} = j) = P(z^{(i)} = j | x^{(i)}; \phi, \mu, \Sigma).$$

Here, "$Q_i(z^{(i)} = j)$" denotes the probability of $z^{(i)}$ taking the value j under the distribution Q_i.

Next, in the M-step, we need to maximize, with respect to our parameters ϕ, μ, Σ, the quantity

$$\begin{aligned}
&\sum_{i=1}^{n} \sum_{z^{(i)}} Q_i(z^{(i)}) \log \frac{p(x^{(i)}, z^{(i)}; \phi, \mu, \Sigma)}{Q_i(z^{(i)})} \\
&\quad = \sum_{i=1}^{n} \sum_{j=1}^{k} Q_i(z^{(i)} = j) \log \frac{p(x^{(i)} | z^{(i)} = j; \mu, \Sigma) p(z^{(i)} = j; \phi)}{Q_i(z^{(i)} = j)} \\
&\quad = \sum_{i=1}^{n} \sum_{j=1}^{k} w_j^{(i)} \log \frac{\frac{1}{(2\pi)^{d/2}|\Sigma_j|^{1/2}} \exp\left(-\frac{1}{2}(x^{(i)} - \mu_j)^T \Sigma_j^{-1} (x^{(i)} - \mu_j)\right) \cdot \phi_j}{w_j^{(i)}}
\end{aligned}$$

Let's maximize this with respect to μ_l. If we take the derivative with respect to μ_l, we find

$$\begin{aligned}
&\nabla_{\mu_l} \sum_{i=1}^{n} \sum_{j=1}^{k} w_j^{(i)} \log \frac{\frac{1}{(2\pi)^{d/2}|\Sigma_j|^{1/2}} \exp\left(-\frac{1}{2}(x^{(i)} - \mu_j)^T \Sigma_j^{-1} (x^{(i)} - \mu_j)\right) \cdot \phi_j}{w_j^{(i)}} \\
&\quad = \quad -\nabla_{\mu_l} \sum_{i=1}^{n} \sum_{j=1}^{k} w_j^{(i)} \frac{1}{2} (x^{(i)} - \mu_j)^T \Sigma_j^{-1} (x^{(i)} - \mu_j) \\
&\quad = \quad \frac{1}{2} \sum_{i=1}^{n} w_l^{(i)} \nabla_{\mu_l} 2 \mu_l^T \Sigma_l^{-1} x^{(i)} - \mu_l^T \Sigma_l^{-1} \mu_l \\
&\quad = \quad \sum_{i=1}^{n} w_l^{(i)} \left(\Sigma_l^{-1} x^{(i)} - \Sigma_l^{-1} \mu_l\right)
\end{aligned}$$

Setting this to zero and solving for μ_l therefore yields the update rule

$$\mu_l := \frac{\sum_{i=1}^{n} w_l^{(i)} x^{(i)}}{\sum_{i=1}^{n} w_l^{(i)}},$$

which was what we had in the previous set of notes.

Let's do one more example, and derive the M-step update for the parameters ϕ_j. Grouping together only the terms that depend on ϕ_j, we find that we need to maximize

$$\sum_{i=1}^{n}\sum_{j=1}^{k} w_j^{(i)} \log \phi_j.$$

However, there is an additional constraint that the ϕ_j's sum to 1, since they represent the probabilities $\phi_j = p(z^{(i)} = j; \phi)$. To deal with the constraint that $\sum_{j=1}^{k} \phi_j = 1$, we construct the Lagrangian

$$\mathcal{L}(\phi) = \sum_{i=1}^{n}\sum_{j=1}^{k} w_j^{(i)} \log \phi_j + \beta(\sum_{j=1}^{k} \phi_j - 1),$$

where β is the Lagrange multiplier.[5] Taking derivatives, we find

$$\frac{\partial}{\partial \phi_j}\mathcal{L}(\phi) = \sum_{i=1}^{n} \frac{w_j^{(i)}}{\phi_j} + \beta$$

Setting this to zero and solving, we get

$$\phi_j = \frac{\sum_{i=1}^{n} w_j^{(i)}}{-\beta}$$

I.e., $\phi_j \propto \sum_{i=1}^{n} w_j^{(i)}$. Using the constraint that $\sum_j \phi_j = 1$, we easily find that $-\beta = \sum_{i=1}^{n}\sum_{j=1}^{k} w_j^{(i)} = \sum_{i=1}^{n} 1 = n$. (This used the fact that $w_j^{(i)} = Q_i(z^{(i)} = j)$, and since probabilities sum to 1, $\sum_j w_j^{(i)} = 1$.) We therefore have our M-step updates for the parameters ϕ_j:

$$\phi_j := \frac{1}{n}\sum_{i=1}^{n} w_j^{(i)}.$$

The derivation for the M-step updates to Σ_j are also entirely straightforward.

[5] We don't need to worry about the constraint that $\phi_j \geq 0$, because as we'll shortly see, the solution we'll find from this derivation will automatically satisfy that anyway.

4 Variational inference and variational auto-encoder

Loosely speaking, variational auto-encoder [2] generally refers to a family of algorithms that extend the EM algorithms to more complex models parameterized by neural networks. It extends the technique of variational inference with the additional "re-parametrization trick" which will be introduced below. Variational auto-encoder may not give the best performance for many datasets, but it contains several central ideas about how to extend EM algorithms to high-dimensional continuous latent variables with non-linear models. Understanding it will likely give you the language and backgrounds to understand various recent papers related to it.

As a running example, we will consider the following parameterization of $p(x, z; \theta)$ by a neural network. Let θ be the collection of the weights of a neural network $g(z; \theta)$ that maps $z \in \mathbb{R}^k$ to $\mathbb{R}^d$. Let

$$z \sim \mathcal{N}(0, I_{k\times k}) \tag{18}$$

$$x|z \sim \mathcal{N}(g(z;\theta), \sigma^2 I_{d\times d}) \tag{19}$$

Here $I_{k\times k}$ denotes identity matrix of dimension k by k, and σ is a scalar that we assume to be known for simplicity.

For the Gaussian mixture models in Section 3, the optimal choice of $Q(z) = p(z|x;\theta)$ for each fixed θ, that is the posterior distribution of z, can be analytically computed. In many more complex models such as the model (19), it's intractable to compute the exact the posterior distribution $p(z|x;\theta)$.

Recall that from equation (10), ELBO is always a lower bound for any choice of Q, and therefore, we can also aim for finding an **approximation** of the true posterior distribution. Often, one has to use some particular form to approximate the true posterior distribution. Let $\mathcal{Q}$ be a family of Q's that we are considering, and we will aim to find a Q within the family of $\mathcal{Q}$ that is closest to the true posterior distribution. To formalize, recall the definition of the ELBO lower bound as a function of Q and θ defined in equation (14)

$$\mathrm{ELBO}(Q,\theta) = \sum_{i=1}^{n} \mathrm{ELBO}(x^{(i)}; Q_i, \theta) = \sum_i \sum_{z^{(i)}} Q_i(z^{(i)}) \log \frac{p(x^{(i)}, z^{(i)}; \theta)}{Q_i(z^{(i)})}$$

Recall that EM can be viewed as alternating maximization of $\mathrm{ELBO}(Q,\theta)$. Here instead, we optimize the EBLO over $Q \in \mathcal{Q}$

$$\max_{Q\in\mathcal{Q}} \max_{\theta} \mathrm{ELBO}(Q,\theta) \tag{20}$$

Now the next question is what form of Q (or what structural assumptions to make about Q) allows us to efficiently maximize the objective above. When the latent variable z are high-dimensional discrete variables, one popular assumption is the **mean field assumption**, which assumes that $Q_i(z)$ gives a distribution with independent coordinates, or in other words, Q_i can be decomposed into $Q_i(z) = Q_i^1(z_1) \cdots Q_i^k(z_k)$. There are tremendous applications of mean field assumptions to learning generative models with discrete latent variables, and we refer to [1] for a survey of these models and their impact to a wide range of applications including computational biology, computational neuroscience, social sciences. We will not get into the details about the discrete latent variable cases, and our main focus is to deal with continuous latent variables, which requires not only mean field assumptions, but additional techniques.

When $z \in \mathbb{R}^k$ is a continuous latent variable, there are several decisions to make towards successfully optimizing (20). First we need to give a succinct representation of the distribution Q_i because it is over an infinite number of points. A natural choice is to assume Q_i is a Gaussian distribution with some mean and variance. We would also like to have more succinct representation of the means of Q_i of all the examples. Note that $Q_i(z^{(i)})$ is supposed to approximate $p(z^{(i)}|x^{(i)}; \theta)$. It would make sense let all the means of the Q_i's be some function of $x^{(i)}$. Concretely, let $q(\cdot; \phi), v(\cdot; \phi)$ be two functions that map from dimension d to k, which are parameterized by ϕ and ψ, we assume that

$$Q_i = \mathcal{N}(q(x^{(i)}; \phi), \mathrm{diag}(v(x^{(i)}; \psi))^2) \tag{21}$$

Here $\mathrm{diag}(w)$ means the $k \times k$ matrix with the entries of $w \in \mathbb{R}^k$ on the diagonal. In other words, the distribution Q_i is assumed to be a Gaussian distribution with independent coordinates, and the mean and standard deviations are governed by q and v. Often in variational auto-encoder, q and v are chosen to be neural networks.[6] In recent deep learning literature, often q, v are called **encoder** (in the sense of encoding the data into latent code), whereas $g(z; \theta)$ if often referred to as the **decoder.**

We remark that Q_i of such form in many cases are very far from a good approximation of the true posterior distribution. However, some approximation is necessary for feasible optimization. In fact, the form of Q_i needs to satisfy other requirements (which happened to be satisfied by the form (21))

Before optimizing the ELBO, let's first verify whether we can efficiently evaluate the value of the ELBO for fixed Q of the form (21) and θ. We

[6] q and v can also share parameters. We sweep this level of details under the rug in this note.

rewrite the ELBO as a function of ϕ, ψ, θ by

$$\text{ELBO}(\phi, \psi, \theta) = \sum_{i=1}^{n} \text{E}_{z^{(i)} \sim Q_i} \left[\log \frac{p(x^{(i)}, z^{(i)}; \theta)}{Q_i(z^{(i)})} \right], \tag{22}$$
$$\text{where } Q_i = \mathcal{N}(q(x^{(i)}; \phi), \text{diag}(v(x^{(i)}; \psi))^2)$$

Note that to evaluate $Q_i(z^{(i)})$ inside the expectation, we should be able **to compute the density of** Q_i. To estimate the expectation $\text{E}_{z^{(i)} \sim Q_i}$, we should be able **to sample from distribution** Q_i so that we can build an empirical estimator with samples. It happens that for Gaussian distribution $Q_i = \mathcal{N}(q(x^{(i)}; \phi), \text{diag}(v(x^{(i)}; \psi))^2)$, we are able to be both efficiently.

Now let's optimize the ELBO. It turns out that we can run gradient ascent over ϕ, ψ, θ instead of alternating maximization. There is no strong need to compute the maximum over each variable at a much greater cost. (For Gaussian mixture model in Section 3, computing the maximum is analytically feasible and relatively cheap, and therefore we did alternating maximization.) Mathematically, let η be the learning rate, the gradient ascent step is

$$\begin{aligned}
\theta &:= \theta + \eta \nabla_\theta \text{ELBO}(\phi, \psi, \theta) \\
\phi &:= \phi + \eta \nabla_\phi \text{ELBO}(\phi, \psi, \theta) \\
\psi &:= \psi + \eta \nabla_\psi \text{ELBO}(\phi, \psi, \theta)
\end{aligned}$$

Computing the gradient over θ is simple because

$$\begin{aligned}
\nabla_\theta \text{ELBO}(\phi, \psi, \theta) &= \nabla_\theta \sum_{i=1}^{n} \text{E}_{z^{(i)} \sim Q_i} \left[\frac{\log p(x^{(i)}, z^{(i)}; \theta)}{Q_i(z^{(i)})} \right] \\
&= \nabla_\theta \sum_{i=1}^{n} \text{E}_{z^{(i)} \sim Q_i} \left[\log p(x^{(i)}, z^{(i)}; \theta) \right] \\
&= \sum_{i=1}^{n} \text{E}_{z^{(i)} \sim Q_i} \left[\nabla_\theta \log p(x^{(i)}, z^{(i)}; \theta) \right], \qquad (23)
\end{aligned}$$

But computing the gradient over ϕ and ψ is tricky because the sampling distribution Q_i depends on ϕ and ψ. (Abstractly speaking, the issue we face can be simplified as the problem of computing the gradient $\text{E}_{z \sim Q_\phi}[f(\phi)]$ with respect to variable ϕ. We know that in general, $\nabla \text{E}_{z \sim Q_\phi}[f(\phi)] \neq \text{E}_{z \sim Q_\phi}[\nabla f(\phi)]$ because the dependency of Q_ϕ on ϕ has to be taken into account as well.)

The idea that comes to rescue is the so-called **re-parameterization trick**: we rewrite $z^{(i)} \sim Q_i = \mathcal{N}(q(x^{(i)}; \phi), \text{diag}(v(x^{(i)}; \psi))^2)$ in an equivalent

way:

$$z^{(i)} = q(x^{(i)}; \phi) + v(x^{(i)}; \psi) \odot \xi^{(i)} \text{ where } \xi^{(i)} \sim \mathcal{N}(0, I_{k\times k}) \tag{24}$$

Here $x \odot y$ denotes the entry-wise product of two vectors of the same dimension. Here we used the fact that $x \sim N(\mu, \sigma^2)$ is equivalent to that $x = \mu + \xi\sigma$ with $\xi \sim N(0, 1)$. We mostly just used this fact in every dimension simultaneously for the random variable $z^{(i)} \sim Q_i$.

With this re-parameterization, we have that

$$\begin{aligned} &\mathrm{E}_{z^{(i)}\sim Q_i}\left[\log \frac{p(x^{(i)}, z^{(i)}; \theta)}{Q_i(z^{(i)})}\right] \\ &= \mathrm{E}_{\xi^{(i)}\sim\mathcal{N}(0,1)}\left[\log \frac{p(x^{(i)}, q(x^{(i)}; \phi) + v(x^{(i)}; \psi) \odot \xi^{(i)}; \theta)}{Q_i(q(x^{(i)}; \phi) + v(x^{(i)}; \psi) \odot \xi^{(i)})}\right] \end{aligned} \tag{25}$$

It follows that

$$\begin{aligned} &\nabla_\phi \mathrm{E}_{z^{(i)}\sim Q_i}\left[\log \frac{p(x^{(i)}, z^{(i)}; \theta)}{Q_i(z^{(i)})}\right] \\ &= \nabla_\phi \mathrm{E}_{\xi^{(i)}\sim\mathcal{N}(0,1)}\left[\log \frac{p(x^{(i)}, q(x^{(i)}; \phi) + v(x^{(i)}; \psi) \odot \xi^{(i)}; \theta)}{Q_i(q(x^{(i)}; \phi) + v(x^{(i)}; \psi) \odot \xi^{(i)})}\right] \\ &= \mathrm{E}_{\xi^{(i)}\sim\mathcal{N}(0,1)}\left[\nabla_\phi \log \frac{p(x^{(i)}, q(x^{(i)}; \phi) + v(x^{(i)}; \psi) \odot \xi^{(i)}; \theta)}{Q_i(q(x^{(i)}; \phi) + v(x^{(i)}; \psi) \odot \xi^{(i)})}\right] \end{aligned}$$

We can now sample multiple copies of $\xi^{(i)}$'s to estimate the the expectation in the RHS of the equation above.[7] We can estimate the gradient with respect to ψ similarly, and with these, we can implement the gradient ascent algorithm to optimize the ELBO over ϕ, ψ, θ.

There are not many high-dimensional distributions with analytically computable density function are known to be re-parameterizable. We refer to [2] for a few other choices that can replace Gaussian distribution.

References

[1] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

[2] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[7] Empirically people sometimes just use one sample to estimate it for maximum computational efficiency.