Comparing Gibbs, EM and SEM for MAP Inference in Mixture Models

Manzil Zaheer
Carnegie Mellon University
manzil@cmu.edu

Micheal Wick
Oracle Labs
michael.wick@oracle.com

Satwik Kottur
Carnegie Mellon University
skottur@andrew.cmu.edu

Jean-Baptiste Tristan
Oracle Labs
jean.baptiste.tristan@oracle.com

Abstract

Classic inference algorithms such as Gibbs sampling and EM often perform well in practice, but selecting between them when scaling a model to large datasets is difficult. In particular, Gibbs sampling is easy to distribute, but difficult to parallelize, while EM is easy parallelize, but difficult to distribute. Remarkably, the relatively obscure stochastic EM (SEM) combines the computational strengths of the two methods, without inheriting any of their weaknesses. Indeed, we highlight these strengths by deriving the three algorithms on a simple, yet representative (of the general class) mixture model. We also demonstrate empirically that the MAP solutions found by the three algorithms are comparable across a wide variety of parameter settings; further, SEM appears more robust to poor initialization.

1 Introduction

Latent variable Bayesian mixture models such as Gaussian mixtures and latent Dirichlet allocation are important for a wide variety of problems, including clustering and text understanding/summarization [4, 2, 1]. Inference in these models is difficult and many existing algorithms are based upon Markov chain Monte Carlo (MCMC) [7], expectation maximization (EM) [3], among others [5, 8]. In practice, it is often difficult to determine which inference algorithm will work best since this depends largely on the task, the data, and the model. Therefore, we must often turn to empirical studies to help understand their strengths and weaknesses.

Furthermore, for latent variable mixture models, Gibbs and EM possess remarkably different computational properties. For example, collapsed Gibbs sampling is sequential and difficult to parallelize, but has a relatively small memory footprint and is easy to distribute because we only need to communicate imputed values. In contrast, EM is easy to parallelize, but difficult to distribute because the dense conditional puts tremendous pressure on communication bandwidth and memory. However, the two algorithms have complementary strengths.

We observe that a relatively esoteric algorithm, stochastic EM (SEM), has the potential to combine the strengths of Gibbs and EM. In particular, SEM replaces the full expectation from the E-step with a sample from it, enabling the subsequent maximization step to operate only on the imputed data and current sufficient statistics. As a result, SEM substantially reduces memory and communication costs (even more than the Gibbs sampler which must also store the latent variables). Furthermore, since SEM is based on EM, it is embarassingly parallel. From a scalable systems perspective SEM has clear computational advantages, but does SEM find good quality solutions to the MAP inference optimization problem in practice?

In this paper, we derive Gibbs, EM, and SEM for a somewhat general and representitive class of latent variable Bayesian mixture models. More importantly, we derive the algorithms for a special instance of the class that highlights the computational advantages of SEM over the other two algorithms. Empirically, we study the performance of these algorithms and find that SEM performs

equally (if not better) than EM and Gibbs across many experimental conditions. Further, we find that SEM may be more robust to poor initialization conditions than either the Gibbs sampler or EM.

2 MAP Inference for Mixture Models

A typical example of a mixture model is the Gaussian mixture in which each data point is assumed to be drawn from one of several Gaussian distributions. Assuming that the variance of these Gaussian distributions is known, the goal is to infer the Gaussian means μ and mixing coefficient π from the data. The membership of a data point to one of the Gaussian distributions is modeled using a latent variable that identifies the appropriate Gaussian. To avoid overfitting, we equip the parameters we wish to infer with priors and perform *maximum a posteriori* (MAP) inference, which is an optimization problem in which we seek a setting to the parameters that maximizes the marginal probability of the data. This gives us the following model:

$$\begin{aligned}\pi &\sim Dirichlet(\alpha)\\ z_i &\sim Cat(\pi)\\ \mu_k &\sim \mathcal{N}(\mu_o, \tau_0)\\ x_i &\sim \mathcal{N}(\mu_{z_i}, \tau_1)\end{aligned}$$

Note there are many ways to extend and modify such a model; the class of mixture models is fairly rich. For example, we could use a different prior for the mean, although we often restrict the choice of priors to ensure conjugacy. The data could be drawn from a different distribution. The data itself could have more structure, for instance, the data points could belong to different sets (or documents), each of which with its own mixing coefficients. The latent variables may be selected using another layer of latent variable as in Pachinko allocation [6]. In general, if we choose distributions in the exponential family and ensure conjugacy, it is reasonable to expect the algorithms we present in this paper to work. A well-known example of a mixture model that has more structure than GMM is LDA, which performs topic modeling. It is a discrete data model, where the data points belong to different sets (documents) each with its own mixing coefficient. Appendix D has details of LDA.

3 Gibbs, EM, and SEM on a Simple Example

In this section we present a pedagogical example that highlights the computational differences between the three algorithms (Gibbs, EM, SEM). We choose an example that is both simple and representative of the general class. Simplicity is important because it makes it much easier to see the computational differences. In particular, we choose a mixture of 1-dimensional Gaussians in which the mixing coefficient π_0 is known, the precision τ of each Gaussian is known, its priors $(\mu_0, \kappa_0\tau)$ are known, and inference must infer the means of the Gaussians μ. That is each datum x_i has an associated latent variable z_i that determines the Gaussian from which it is drawn. The sufficient statistics comprise just a single array of size K (one element per component). More formally, we have the following generative model:

$$\begin{aligned}\mu_k &\sim \mathcal{N}(\mu_0, \kappa_0\tau)\\ z_i &\sim \mathsf{Categorical}(\pi_0)\\ x_i &\sim \mathcal{N}(\mu_{z_i}, \tau)\end{aligned}$$

We present the three algorithms in Algorithm 1,2,3. First, note that all three algorithms must store the data in an array x of size n. In addition to x, the Gibbs sampler requires storing the latent variables in an array z and parameters μ of size (n and k respectively). EM requires substantially more memory since an implementation must have four arrays x, μ, v, w of size n, k, $n \times k$, k respectively. Finally, note that SEM has by far the smallest memory requirement since its four arrays x, μ, tmp, w are of size n, k, k, k.

Algorithm 1 Collapsed Gibbs Sampler (CGS)

initialize z randomly and accordingly μ and n
for each iteration $t = 1 \dots T$ **do**
  for each data point $i = 1 \dots N$ **do**
    $j = z_i$
    $\mu_j - = x_i$
    $n_j - = 1$
    for each component $k = 1 \dots K$ **do**
      $p[k] = \pi_k \times \mathcal{N}\left(x_i | \frac{\kappa_0 \mu_0 + \mu_k}{\kappa_0 + n_k}, \tau \frac{\kappa_0 + n_k}{\kappa_0 + n_k + 1}\right)$
    end for
    sample j from p
    $n_j + = 1$
    $\mu_j + = x_i$
    $z_i = j$
  end for
end for
for each component $k = 1 \dots K$ **do**
  $\mu_k = \frac{\kappa_0 \mu_0 + \mu_k}{\kappa_0 + n_k}$
end for

Algorithm 2 EM

initialize z randomly and accordingly μ and n
for each iteration $i = 1 \dots T$ **do**
  for each data point $i = 1 \dots N$ (E-step) **do**
    for each component $k = 1 \dots K$ **do**
      $q_{ik} = \frac{\pi_k \times \mathcal{N}(x_i | \mu_k, \tau)}{\sum_{k'} \pi_k \times \mathcal{N}(x_i | \mu_{k'}, \tau)}$
      $n_k + = q_{ik}$
    end for
  end for
  clear all μ to $\kappa_0 \mu_0$
  for each data point $i = 1 \dots N$ (M-step) **do**
    for each component $k = 1 \dots K$ **do**
      $\mu_k + = q_{ik} \times x_i$
    end for
  end for
  for each component $k = 1 \dots K$ **do**
    $\mu_k = \mu_k / (\kappa_0 + n_k)$
  end for
end for

Algorithm 3 SEM

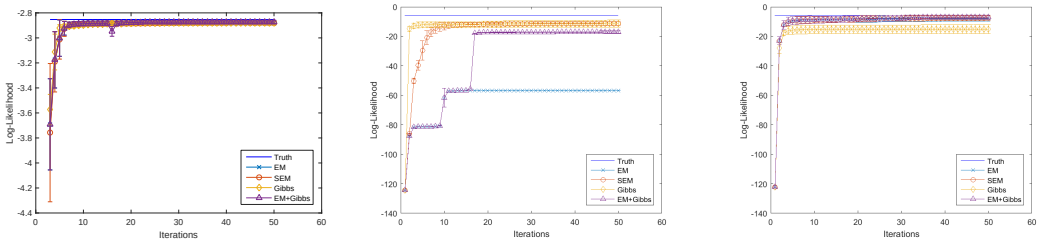
initialize z randomly and accordingly μ and n
for each iteration $i = 1 \dots T$ (SE-step) **do**
  clear tmp
  for each data point $i = 1 \dots N$ **do**
    for each component $k = 1 \dots K$ **do**
      $p[k] = \pi_k \times \mathcal{N}(x_i | \mu_k, \tau)$
    end for
    sample k from p
    $\text{tmp}_k + = x_i$
    $n_k + = 1$
  end for
  for each component $k = 1 \dots K$ (M-step) **do**
    $\mu_k = \frac{\kappa_0 \mu_0 + \text{tmp}_k}{\kappa_0 + n_k}$
  end for
end for

4 Experiments

For the simple example in the previous section, we take $K = 10$ clusters and $n = 100,000$ training examples generated synthetically to remove any error caused by an incorrect model choice

$$\mu_0 = 0 \quad \kappa_0 = 0.1$$
$$\tau = 1 \quad \pi \sim \text{Dirichlet}(1).$$

We compare Gibbs, EM and SEM ($T = 50$ iterations) under several experimental conditions. Further, we include an ad-hoc method in which the first 15 iterations are EM and then the rest are collapsed Gibbs sampling (EM+Gibbs). We repeat each condition 10 times using random initialization and report the average log-likelihood on a held-out test set of 1000 instances. First, as a control, in Figure 1a we run all three algorithms on a simple one-dimensional model. As expected, they perform equally well. Next, we study the robustness of each algorithm to initialization in a two-dimensional model. For this, we choose two extreme initialization conditions under which the algorithms are likely to get stuck in poor local optima. In the first case we initialize to a low entropy configuration (Figure 1b), and the second case a high entropy configuration (Figure 1c). The former causes trouble for EM because it is deterministic; in contrast, the stochasticity in Gibbs and SEM allow them to be robust to this condition. The latter causes trouble for Gibbs because the conditional distribution has high entropy forcing the sampler to rely on sheer luck in the early iterations. In summary, we find SEM is more robust than either algorithm since it works well in both extremes.



(a) Simple 1D case

(b) Initialization to single component

(c) Uniformly random initialization

Figure 1: Simple Cases

We also compare the algorithms on the more general problem of estimating both π and μ under a variety of experimental conditions by varying: the number of training instances N_{train}, number of clusters N_k, and dimensionality of the training instances N_d. For each condition we evaluate on a common test set with $N_{test} = 1000$ instances. More precisely, our experimental conditions include:

- Vary number of mixture components: For each run, we select $N_K \in \{10, 20, 40, 80, 160, 320\}$, keeping the other parameters fixed i.e. $(N_K, N_test) = (10, 1000)$. However, we vary the number of training points to make $\frac{N_{train}}{N_K}$ to be a constant. Hence, we take $N_{train} = 100 * N_K$.
- Vary number of Dimensions: For each run, we select $N_d \in \{1, 2, 4, 8, 16\}$, keeping the other parameters fixed i.e. $(N_K, N_test) = (10, 1000)$. However, we vary the number of training points to make $\frac{N_{train}}{N_D}$ to be a constant. Hence, we take $N_{train} = 1k * N_D$.
- Vary number of training examples:For each run, we select $N_train \in \{1k, 2k, 4k, 8k, 16k, 32k\}$, keeping the other parameters fixed i.e. $(N_d, N_K, N_test) = (2, 10, 1000)$.

We run the three experiments each using the collapsed Gibbs sampler, EM and SEM, each for $T = 50$ iterations. Further, we try an ad-hoc method where first 15 iterations carried out using EM and then we start using collapsed Gibbs sampling. We repeat the experiments 10 times using various random initializations. The log-likelihood (mean and std-error thereof) on a held-out test set of size $1,000$ at the end of 50 iterations is depicted in Figure 2. We find that SEM consistently outperforms all other.

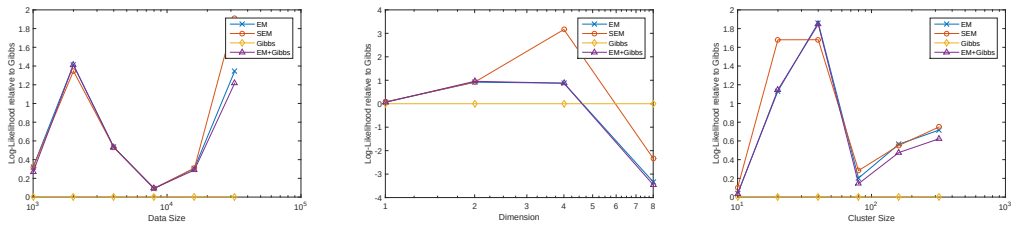


Figure 2: Varying parameters

5 Conclusion

We compared classic inference algorithms on a few mixture models and found that their performance to be comparable, even as the number of components, data size, and dimensions vary. This is especially encouraging because from a computational perspective, SEM is an ideal inference algorithm for mixture models with latent variables. It combines the strengths of EM and Gibbs sampling making it easy to parallelize and distribute. Therefore, we recommend practitioners consider SEM as a foundation upon which to build the next generation of scalable inference algorithms.

References

[1] A. Ahmed, L. Hong, and A.J. Smola. Nested chinese restaurant franchise processes: Applications to user tracking and document modeling. In *ICML*, Atlanta, GA, 2013.

[2] Amr Ahmed, Moahmed Aly, Joseph Gonzalez, Shravan Narayanamurthy, and Alexander J Smola. Scalable inference in latent variable models. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 123–132. ACM, 2012.

[3] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

[4] Nan Du, Mehrdad Farajtabar, Amr Ahmed, Alexander J Smola, and Le Song. Dirichlet-hawkes processes with applications to clustering continuous-time document streams. In *KDD*. ACM, 2015.

[5] Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. An introduction to variational methods for graphical models. *Mach. Learn.*, 37(2):183–233, November 1999.

[6] Wei Li and Andrew McCallum. Pachinko allocation: Scalable mixture models of topic correlations. *J. of Machine Learning Research. Submitted*, 2008.

[7] Radford M Neal. Probabilistic inference using markov chain monte carlo methods. 1993.

[8] Radford M Neal. Mcmc using hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 2, 2011.

A (Stochastic) EM in General

Expectation-Maximization (EM) is an iterative method for finding the maximum likelihood or *maximum a posteriori* (MAP) estimates of the parameters in statistical models when data is only partially, or when model depends on unobserved latent variables. This section is inspired from `http://www.ece.iastate.edu/~namrata/EE527_Spring08/emlecture.pdf`

We derive EM algorithm for a very general class of model. Let us define all the quantities of interest.

Table 1: Notation

Symbol	Meaning
$\mathbf{x}$	Observed data
$\mathbf{z}$	Unobserved data
$(\mathbf{x}, \mathbf{z})$	Complete data
$f_{\mathbf{X};\eta}(\mathbf{x};\eta)$	marginal observed data density
$f_{\mathbf{Z};\eta}(\mathbf{z};\eta)$	marginal unobserved data density
$f_{\mathbf{X},\mathbf{Z};\eta}(\mathbf{x},\mathbf{z};\eta)$	complete data density/likelihood
$f_{\mathbf{Z}\|\mathbf{X};\eta}(\mathbf{z}\|\mathbf{x};\eta)$	conditional unobserved-data (missing-data) density.

Objective: To maximize the marginal log-likelihood or posterior, i.e.

$$L(\eta) = \log f_{\mathbf{X};\eta}(\mathbf{x};\eta). \tag{1}$$

Assumptions:

1. z_i are independent given η. So

$$f_{\mathbf{Z};\eta}(\mathbf{z};\eta) = \prod_{i=1}^{N} f_{Z_i;\eta}(z_i;\eta), \tag{2}$$

2. x_i are independent given missing data z_i and η. So

$$f_{\mathbf{X},\mathbf{Z};\eta}(\mathbf{x},\mathbf{z};\eta) = \prod_{i=1}^{N} f_{X_i,Z_i;\eta}(x_i,z_i;\eta). \tag{3}$$

As a consequence we obtain:

$$f_{\mathbf{Z}|\mathbf{X};\eta}(\mathbf{z}|\mathbf{x};\eta) = \prod_{i=1}^{N} f_{Z_i|X_i;\eta}(z_i|x_i;\eta), \tag{4}$$

Now,

$$L(\eta) = \log f_{\mathbf{X};\eta}(\mathbf{x};\eta) = \log f_{\mathbf{X},\mathbf{Z};\eta}(\mathbf{x},\mathbf{z};\eta) - \log f_{\mathbf{Z}|\mathbf{X};\eta}(\mathbf{z}|\mathbf{x};\eta) \tag{5}$$

or, summing across observations,

$$L(\eta) = \sum_{i=1}^{N} \log f_{X_i;\eta}(x_i;\eta) = \sum_{i=1}^{N} \log f_{X_i,Z_i;\eta}(x_i,z_i;\eta) - \sum_{i=1}^{N} \log f_{Z_i|X_i;\eta}(z_i|x_i;\eta). \tag{6}$$

Let us take the expectation of the above expression with respect to $f_{Z_i|X_i;\eta}(z_i|x_i;\eta_p)$, where we choose $\eta = \eta_p$:

$$\begin{aligned}
&\sum_{i=1}^{N} \mathbb{E}_{Z_i|X_i;\eta}\left[\log f_{X_i;\eta}(x_i;\eta)|x_i;\eta_p\right] \\
&\quad = \sum_{i=1}^{N} \mathbb{E}_{Z_i|X_i;\eta}\left[\log f_{X_i,Z_i;\eta}(x_i,z_i;\eta)|x_i;\eta_p\right] - \sum_{i=1}^{N} \mathbb{E}_{Z_i|X_i;\eta}\left[\log f_{Z_i|X_i;\eta}(z_i|x_i;\eta)|x_i;\eta_p\right]
\end{aligned} \tag{7}$$

Since $L(\eta) = \log f_{\mathbf{X};\eta}(\mathbf{x};\eta)$ does not depend on $\mathbf{z}$, it is invariant for this expectation. So we recover:

$$\begin{aligned} L(\eta) &= \sum_{i=1}^{N} \mathbb{E}_{Z_i|X_i;\eta} \left[\log f_{X_i,Z_i;\eta}(x_i, z_i; \eta)|x_i; \eta_p\right] - \sum_{i=1}^{N} \mathbb{E}_{Z_i|X_i;\eta} \left[\log f_{Z_i|X_i;\eta}(z_i|x_i;\eta)|x_i;\eta_p\right] \\ &= Q(\eta|\eta_p) - H(\eta|\eta_p). \end{aligned} \tag{8}$$

Now, (8) may be written as

$$Q(\eta|\eta_p) = L(\eta) + \underbrace{H(\eta|\eta_p)}_{\leq H(\eta_p|\eta_p)} \tag{9}$$

Here, observe that $H(\eta|\eta_p)$ is maximized (with respect to η) by $\eta = \eta_p$, i.e.

$$H(\eta|\eta_p) \leq H(\eta_p|\eta_p) \tag{10}$$

Simple proof using Jensen's inequality.

As our objective is to maximize $L(\eta)$ with respect to η, if we maximize $Q(\eta|\eta_p)$ with respect to η, it will force $L(\eta)$ to increase. This is what is done repetitively in EM. To summarize, we have:

E-step : Compute $f_{Z_i|X_i;\eta}(z_i|x_i;\eta_p)$ using current estimate of $\eta = \eta_p$.

M-step : Maximize $Q(\eta|\eta_p)$ to obtain next estimate η_{p+1}.

Now assume that the complete data likelihood belongs to the exponential family, i.e.

$$f_{X_i,Z_i;\eta}(x_i, z_i; \eta) = \exp\{T\{z_i \cdot x_i\}\eta - g(\eta)\} \tag{11}$$

then

$$\begin{aligned} Q(\eta|\eta_p) &= \sum_{i=1}^{N} \mathbb{E}_{Z_i|X_i;\eta} \left[\log f_{X_i,Z_i;\eta}(x_i, z_i; \eta)|x_i; \eta_p\right] \\ &= \sum_{i=1}^{N} \mathbb{E}_{Z_i|X_i;\eta} \left[T\{z_i, \cdot, x_i\}\eta g(\eta)|x_i; \eta_p\right] \end{aligned} \tag{12}$$

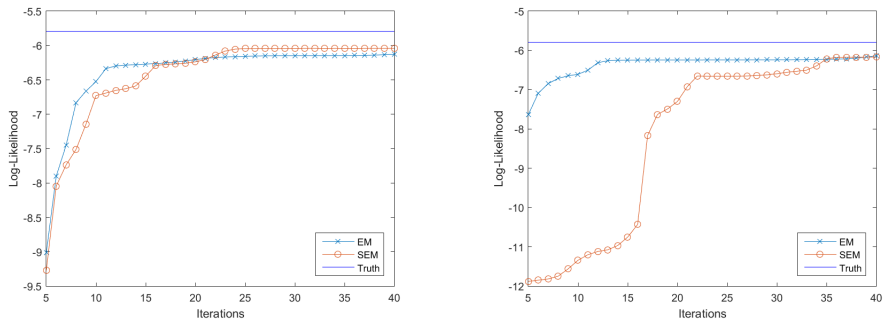
To find the maximizer, differentiate and set it to zero:

$$\frac{1}{N} \sum_{i} \mathbb{E}_{Z_i|X_i;\eta} \left[\{Tz_i, x_i\}\eta|x_i; \eta_p\right] = \frac{dg(\eta)}{d\eta} \tag{13}$$

and one can obtain the maximizer by solving this equation.

Stochastic EM (SEM) introduces an additional simulation after the E-step that replaces the full distribution with a single sample:

S-step Sample $z_i \sim f_{Z_i|X_i;\eta}(z_i|x_i;\eta_p)$



(a) Same initialization (b) Bad initialization for SEM

Figure 3: Performance of SEM

This essentially means we replace $\mathbb{E}[\cdot]$ with an empirical estimate. Thus, instead of solving (13), we simply have:

$$\frac{1}{N}\sum_i T(z_i, x_i) = \frac{dg(\eta)}{d\eta}. \tag{14}$$

Computing and solving this system of equations is considerably easier than (13).

Now to demonstrate that SEM is well behaved and works in practice, we run a small experiment. Consider the problem of estimating the parameters of a Gaussian mixture. We choose a 2-dimensional Gaussian with $K = 30$ clusters and 100,000 training points and 1,000 test points. We run EM and SEM with the following initialization:

- Both SEM and EM are provided the same initialization.
- SEM is deliberately provided a bad initialization, while EM is not.

The log-likelihood on the heldout test set is shown in Figure 3.

B (S)EM Derivation for GMM

Endless flow of equations The EM iteration alternates between performing an expectation (E) step, which creates a function for the expectation of the log-likelihood (ℓ) evaluated using the current estimate for the parameters (initially, random values), and a maximization (M) step, which computes parameters maximizing the expected log-likelihood found on the E step. These parameter-estimates are then used to determine the distribution of the latent variables in the next E step.

We are in a Bayesian world, so parameters are treated as Random Variables:

$$\begin{aligned}
\log p(X, \theta, \pi | m_0, \phi_0, \alpha) &= \log \sum_Z p(X, Z, \theta, \pi | m_0, \phi_0, \alpha) \\
&= \log \sum_Z p(X, Z | \theta, \pi) + \sum_k \log p(\theta_k | m_0, \phi_0) + \log p(\pi | \alpha) \\
&= \sum_i \log \sum_k p(x_i, z_i = k | \theta, \pi) + \sum_k \log p(\theta_k | m_0, \phi_0) + \log p(\pi | \alpha) \\
&= \sum_i \log \sum_k \frac{q(z_i = k | x_i)}{q(z_i = k | x_i)} p(x_i, z_i = k | \theta, \pi) + \sum_k \log p(\theta_k | m_0, \phi_0) + \log p(\pi | \alpha) \\
&= \sum_i \log \sum_k q(z_i = k | x_i) \frac{p(x_i, z_i = k | \theta, \pi)}{q(z_i = k | x_i)} + \sum_k \log p(\theta_k | m_0, \phi_0) + \log p(\pi | \alpha) \\
&\geq \sum_i \sum_k q(z_i = k | x_i) \log \frac{p(x_i, z_i = k | \theta, \pi)}{q(z_i = k | x_i)} + \sum_k \log p(\theta_k | m_0, \phi_0) + \log p(\pi | \alpha)
\end{aligned} \tag{15}$$

Let's denote

$$F(q, \theta, \pi) = \sum_i \sum_k q(z_i = k | x_i) \log \frac{p(x_i, z_i = k | \theta, \pi)}{q(z_i = k | x_i)} + \sum_k \log p(\theta_k | m_0, \phi_0) + p(\pi | \alpha)$$

EM algorithm (in coordinate descent manner) works as follows:

- In E-step, fix θ and π, maximize F over q. On re-arranging one could get:

$$F(q, \theta, \pi) = -\sum_i D_{KL}(q(z_i | x_i) || P(z_i | x_i, \theta, \pi)) + \log P(x_i | \theta, \pi) + \sum_k \log p(\theta_k | m_0, \phi_0) + p(\pi | \alpha)$$

D_{KL} denotes the Kullback Leibler divergence, a measure of the divergence of two distributions, which is defined as $D_{KL}(P||Q) = \sum_i P(i) \ln \frac{P(i)}{Q(i)}$. Since in E-step, θ and π are fixed, $F(q, \theta, \pi)$ is maximized only by q that maximizes the negative KL divergence. Because KL divergence is always non-negative. $D_{KL} = 0$ happens only when p and q are the same.

Summary: In the E-step of t^{th} iteration, we derive $q^{(t)} = \text{argmax}_q F(q, \theta^{(t-1)}, \pi^{(t-1)})$, namely

$$\begin{aligned}
n_{ik} = E_{q|x}[z_i = k] = q^{(t)}(z_i = k | x_i) &= p(z_i = k | x_i; \theta^{(t-1)}, \pi^{t-1}) \\
&\propto p(x_i | \theta_k^{t-1}, z_i = k) p(z_i = k | \pi^{t-1}) \\
&= \frac{\pi_k^{t-1} p(x_i | \theta_k^{t-1})}{\sum_{k'} \pi_{k'}^{t-1} p(x_i | \theta_{k'}^{t-1})}
\end{aligned} \tag{16}$$

- In M-step, fix q, maximize F over θ and π.

We begin by maximizing over θ, in which case we can drop other terms:

$$\begin{aligned}
F(q, \theta, \pi) &= \sum_i \sum_k q(z_i = k | x_i) \log p(x_i, z_i = k | \theta, \pi) + \sum_k \log p(\theta_k | m_0, \phi_0) + \text{const} \\
&= \sum_i \sum_k q(z_i = k | x_i) (\langle \phi(x_i), \theta_k \rangle - g(\theta_k)) + \sum_k (\langle \phi_0, \theta_k \rangle - m_0 g(\theta_k) + \text{const}
\end{aligned} \tag{17}$$

Taking derivative with respect to θ_k and setting it to 0 yields

$$\begin{aligned}\nabla g(\hat{\theta}_k) &= \frac{1}{m_0 + \sum_k q(z_i = k|x_i)} \left(\phi_0 + \sum_i q(z_i = k|x_i)\phi(x_i) \right) \\ \hat{\theta}_k &= \eta^{-1} \left(\frac{1}{m_0 + \sum_k q(z_i = k|x_i)} \left(\phi_0 + \sum_i q(z_i = k|x_i)\phi(x_i) \right) \right)\end{aligned} \tag{18}$$

Similarly solving for π, we first summarize the equation with the terms related to π_k as follwoing:

$$\begin{aligned}F(q, \theta, \pi) &= \sum_i \sum_k q(z_i = k|x_i) \log p(x_i, z_i = k|\theta, \pi) + \sum_k \log p(\theta_k|m_0, \phi_0) + \log p(\pi|\alpha) \\ F(q, \theta, \pi) &= \sum_i \sum_k q(z_i = k|x_i) \log \pi_k + \sum_k (\alpha_k - 1) \log \pi_k + const\end{aligned} \tag{19}$$

Now solving over π_k leads to solving the following optimization function,

$$\begin{aligned}\hat{\pi} &= \underset{\pi}{\operatorname{argmax}} \sum_i \sum_k q(z_i = k|x_i) \log \pi_k + \sum_k (\alpha_k - 1) \log \pi_k \\ \text{s.t.} \quad & \sum_{k=1}^{K} \pi_k = 1.\end{aligned} \tag{20}$$

Writing the lagrangian function for the given optimization function,

$$L(\pi, \lambda) = \sum_i \sum_k q(z_i = k|x_i) \log \pi_k + \sum_k (\alpha_k - 1) \log \pi_k + \lambda(1 - \sum_{k=1}^{K} \pi_k) \tag{21}$$

Now, setting the gradient with respect to π_k gives us

$$\begin{aligned}0 &= \frac{1}{\hat{\pi_k}} \sum_i [q(z_i = k|x_i) + (\alpha_k - 1)] + \lambda \\ \Leftrightarrow \quad \hat{\pi_k} &= \frac{\sum_i q(z_i = k|x_i) + (\alpha_k - 1)}{\lambda}\end{aligned} \tag{22}$$

Since $q(z_i = k|x_i) + (\alpha_k - 1) \geq 0$ and π_k have to sum up to 1, solving for λ, and thereby obtaining the solution for π_k as

$$\begin{aligned}\hat{\pi}_k &= \frac{\sum_i q(z_i = k|x_i) + \alpha_k - 1}{\sum_{i,k} q(z_i = k|x_i) + \sum_k \alpha_k - k} \\ &= \frac{\sum_i q(z_i = k|x_i) + \alpha_k - 1}{N + \sum_k \alpha_k - K}.\end{aligned} \tag{23}$$

The distribution of Multivariate Normal $\mathcal{N}(\mu, \Sigma)$ is given by

$$p(x|\mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^d |\Sigma|}} \exp\left(-\frac{1}{2}(x - \mu)^{\mathrm{T}} \Sigma^{-1} (x - \mu) \right) \tag{24}$$

where $\mu \in \mathbb{R}^d$ and $\Sigma \succ 0$ is a symmetric positive definite $d \times d$ matrix.

The conjugate prior for Multivariate Normal Distribution can be parametrized as the Normal Inverse Wishart Distribution $\mathcal{NIW}(\mu_0, \kappa_0, \Sigma_0, \nu_0)$. The distribution is given by:

$$\begin{aligned}p(\mu, \Sigma;\ \mu_0, \kappa_0, \Sigma_0, \nu_0) &= \mathcal{N}(\mu|\mu_0, \Sigma/\kappa_0)\mathcal{W}^{-1}(\Sigma|\Sigma_0, \nu_0) \\ &= \frac{\kappa_0^{\frac{d}{2}} |\Sigma_0|^{\frac{\nu_0}{2}} |\Sigma|^{-\frac{\nu_0 + d + 2}{2}}}{2^{\frac{(\nu_0+1)d}{2}} \pi^{\frac{d}{2}} \Gamma_d(\frac{\nu_0}{2})} e^{-\frac{\kappa_0}{2}(\mu - \mu_0)^{\mathrm{T}} \Sigma^{-1} (\mu - \mu_0) - \frac{1}{2} \operatorname{tr}(\Sigma_0 \Sigma^{-1})}\end{aligned} \tag{25}$$

Now, derive the Expectation-Maximiazation rules for the mixture of Mutivariate Normal, $\mathcal{N}(\mu_k, \Sigma_k)$ for $k = 1, \ldots, K$ and with the shared prior $\mathcal{NIW}(\mu_0, \kappa_0, \Sigma_0, \nu_0)$.

E step:

$$n_{ik}^{(t)} = \frac{\exp\left[-\frac{1}{2}(x_i - \mu_k^{(t)})^\top \Sigma_k^{(t)-1}(x_i - \mu_k^{(t)})\right]\pi_k^{(t)}}{\sum_{j=1}^k \exp\left[-\frac{1}{2}(x_i - \mu_j^{(t)})^\top \Sigma_j^{(t)-1}(x_i - \mu_j^{(t)})\right]\pi_j^{(t)}} \tag{26}$$

M step: First,

$$\pi_k^{(t+1)} = \frac{\alpha_k - 1 + \sum_i n_{ik}^{(t)}}{N + \sum_{j=1}^K \alpha_j - K}. \tag{27}$$

Now, in natural parameter space, $\theta_1 = \Sigma^{-1}\mu$ and $\theta_2 = -\frac{1}{2}\Sigma^{-1}$. Thus,

$$\begin{aligned} \Sigma &= -\frac{1}{2}\theta_2^{-1} \\ \mu &= -\frac{1}{2}\theta_2^{-1}\theta_1 \end{aligned} \tag{28}$$

and

$$g(\theta) = \begin{bmatrix} -\frac{1}{4}\theta_1^\top \theta_2^{-1}\theta_1 \\ \frac{d}{2}\log(2\pi) - \frac{1}{2}\log|-2\theta_2| \end{bmatrix}. \tag{29}$$

So

$$\frac{\partial g}{\partial \theta_1} = \begin{bmatrix} -\frac{1}{2}\theta_2^{-1}\theta_1 \\ 0 \end{bmatrix} = \begin{bmatrix} \mu \\ 0 \end{bmatrix} \qquad \frac{\partial g}{\partial \theta_2} = \begin{bmatrix} \frac{1}{4}\theta_2^{-1}\theta_1\theta_1^\top\theta_2^{-1} \\ -\frac{1}{2}\theta_2^{-1} \end{bmatrix} = \begin{bmatrix} \mu\mu^\top \\ \Sigma \end{bmatrix}. \tag{30}$$

The derivative $\frac{\partial g_1}{\partial \theta_2}$ comes from the identity $\frac{\partial tr(X^{-1}A)}{\partial X} = -(X^{-1})^\top A (X^{-1})^\top$ and the invariance of the trace operator under cyclic permutation; see 'Wikipedia/Matrix calculus'. Also recall that

$$\phi_0 = \begin{pmatrix} \kappa_0\mu_0 \\ \Sigma_0 + \kappa_0\mu_0\mu_0^\top \end{pmatrix} \qquad m_0 = \begin{pmatrix} \kappa_0 \\ \nu_0 + d + 2 \end{pmatrix} \qquad \phi(x) = \begin{pmatrix} x \\ xx^\top \end{pmatrix}. \tag{31}$$

Denote $n_k^{(t)} = \sum_i n_{ik}^{(t)}$ Combining all of this with (17) we see that

$$\begin{aligned} \frac{\partial F}{\partial \theta_1} &= \kappa_0\mu_0 + \sum_i n_{ik}^{(t)} x_i - \left(\kappa_0 + n_k^{(t)}\right)\mu_k^{(t+1)} = 0 \\ \frac{\partial F}{\partial \theta_2} &= \Sigma_0 + \kappa_0\mu_0\mu_0^\top + \sum_i n_{ik}^{(t)} x_i x_i^\top - \left(\kappa_0 + n_k^{(t)}\right)\mu_k^{(t+1)}\mu_k^{(t+1)\top} - \left(\nu_0 + d + 2 + n_k^{(t)}\right)\Sigma_k^{(t+1)} = 0 \end{aligned} \tag{32}$$

and hence

$$\begin{aligned} \mu_k^{(t+1)} &= \frac{\kappa_0\mu_0 + \sum_i n_{ik}^{(t)} x_i}{\kappa_0 + n_k^{(t)}} \\ \Sigma_k^{(t+1)} &= \frac{\Sigma_0 + \kappa_0\mu_0\mu_0^\top + \sum_i n_{ik}^{(t)} x_i x_i^\top - \left(\kappa_0 + n_k^{(t)}\right)\mu_k^{(t+1)}\mu_k^{(t+1)\top}}{\nu_0 + d + 2 + n_k^{(t)}}. \end{aligned} \tag{33}$$

B.1 Introducing Stochasticity

After performing the E-step, we add an extra simulation step, i.e. we draw and impute the values for the latent variables from its distribution conditioned on data and current estimate of the parameters. This means basically n_{ik} gets transformed into $\delta(z_i - \tilde{k})$ where $\tilde{k}$ is value drawn from the conditional distribution. Then we proceed to perform the M-step, which is even simpler now. To summarize SEM for GMM will have following steps:

E-step in parallel compute the conditional distribution locally:

$$n_{ik}^{(t)} = \frac{\exp\left[-\frac{1}{2}(x_i - \mu_k^{(t)})^\top \Sigma_k^{(t)-1}(x_i - \mu_k^{(t)})\right]\pi_k^{(t)}}{\sum_{j=1}^k \exp\left[-\frac{1}{2}(x_i - \mu_j^{(t)})^\top \Sigma_j^{(t)-1}(x_i - \mu_j^{(t)})\right]\pi_j^{(t)}} \tag{34}$$

S-step in parallel draw z_i from the categorical distribution:

$$z_i^{(t)} \sim \mathsf{Categorical}(n_{i1}^{(t)}, ..., n_{iK}^{(t)}) \tag{35}$$

M-step in parallel compute the new parameter estimates:

$$\begin{aligned}
\pi_k^{(t+1)} &= \frac{\alpha_k - 1 + T_k^{(t)}}{N + \sum_{j=1}^{K} \alpha_j - K} \\
\mu_k^{(t+1)} &= \frac{\kappa_0 \mu_0 + \sum_{i|z_i^{(t)}=k} x_i}{\kappa_0 + T_k^{(t)}} \\
\Sigma_k^{(t+1)} &= \frac{\Sigma_0 + \kappa_0 \mu_0 \mu_0^\top + \sum_{i|z_i^{(t)}=k} x_i x_i^\top - \left(\kappa_0 + T_k^{(t)}\right) \mu_k^{(t+1)} \mu_k^{(t+1)\top}}{\nu_0 + d + 2 + n_k^{(t)}}.
\end{aligned} \tag{36}$$

where $T_k^{(t)} = \left| \left\{ z_i^{(t)} \mid z_i^{(t)} = k \right\} \right|$.

C Gibbs Sampler Derivation for GMM

The Markov blanket for z_i becomes $x_{-i}, z_{-i}, x_i, \phi_0, m_0$, and α in this case. We obtain

$$\begin{aligned} P(z_i \mid rest) &= \frac{P(x_i \mid z_i, \{x_j : z_j = z_i\}, m_0, \phi_0) P(z_i \mid z_{-i}, \alpha)}{P(x_i \mid x_{-i}, m_0, \phi_0)} \\ &= \frac{\exp[h(m_{z_i}, \phi_{z_i}) - h(m_{z_i} - 1, \phi_{z_i} - \phi(x_i))] \exp[\log \mathbf{B}(\vec{n_k}) - \log \mathbf{B}(\vec{n_k} - \vec{e_{z_i}})]}{\sum_{j=1}^{K} \exp[h(m_j, \phi_j) - h(m_j - 1, \phi_j - \phi(x_i))] \exp[\log \mathbf{B}(\vec{n_k}) - \log \mathbf{B}(\vec{n_k} - \vec{e_j})]} \\ &= \frac{\exp[h(m_{z_i}, \phi_{z_i}) - h(m_{z_i} - 1, \phi_{z_i} - \phi(x_i))] \frac{n_{z_i} - 1}{(\sum_k n_k) - 1}}{\sum_{j=1}^{K} \exp[h(m_j, \phi_j) - h(m_j - 1, \phi_j - \phi(x_i))] \frac{n_j - 1}{(\sum_k n_k) - 1}} \\ &= \frac{\exp[h(m_{z_i}, \phi_{z_i}) - h(m_{z_i} - 1, \phi_{z_i} - \phi(x_i))](n_{z_i} - 1)}{\sum_{j=1}^{K} \exp[h(m_j, \phi_j) - h(m_j - 1, \phi_j - \phi(x_i))](n_j - 1)} \end{aligned} \tag{37}$$

which yields the conditional needed in Gibbs sampling to sample a latent variable z_i.

As a pedagogical example, we derive Gibbs sampler for the Multivariate Gaussian with a Normal Inverse Wishart.

First, apply the downdate equations in the following order:

$$\begin{aligned} \kappa_{z_i} &\leftarrow \kappa_{z_i} - 1, \qquad \nu_{z_i} \leftarrow \nu_{z_i} - 1 \\ \mu_{z_i} &\leftarrow \frac{(\kappa_{z_i} - 1)\mu_{z_i}}{\kappa_{z_i}} - x_i \\ \Sigma_{z_i} &\leftarrow \Sigma_{z_i} - \frac{\kappa_{z_i}}{\kappa_{z_i} - 1}(x_i - \mu_{z_i})(x_i - \mu_{z_i})^\top \end{aligned} \tag{38}$$

Update z_i by smpling from the distribution:

$$\begin{aligned} P(z_i = k \mid rest) &= \frac{P(x_i \mid z_i = k, x_{-i})(n_k - 1)}{\sum_j P(x_i \mid z_i = j, x_{-i})(n_j - 1)} \\ &= \frac{t_{\nu_k - d + 1}\left(x_i \;\middle|\; \mu_k, \frac{(\kappa_k + 1)\Sigma_k}{\kappa_k(\nu_k - d + 1)}\right)(n_k - 1)}{\sum_{j=1}^{K} t_{\nu_j - d + 1}\left(x_i \;\middle|\; \mu_j, \frac{(\kappa_j + 1)\Sigma_j}{\kappa_j(\nu_j - d + 1)}\right)(n_j - 1)} \end{aligned} \tag{39}$$

then apply the update equations in the following order:

$$\begin{aligned} \Sigma_{z_i} &\leftarrow \Sigma_{z_i} + \frac{\kappa_{z_i}}{\kappa_{z_i} - 1}(x_i - \mu_{z_i})(x_i - \mu_{z_i})^\top \\ \mu_{z_i} &\leftarrow \frac{\kappa_{z_i}\mu_{z_i} + x_i}{\kappa_{z_i} + 1} \\ \nu_{z_i} &\leftarrow \nu_{z_i} + 1, \qquad \kappa_{z_i} \leftarrow \kappa_{z_i} + 1 \end{aligned} \tag{40}$$

At the end of the procedure, we obtain

$$\begin{aligned} \widehat{\mu}_k &= \frac{\kappa_0\mu_0 + \sum_{i:z_i=k} x_i}{\kappa_0 + \tilde{n}_k} \\ \widehat{\Sigma}_k &= \frac{\Sigma_0 + \sum_{i:z_i=k} x_i x_i^\top + \kappa_0\mu_0\mu_0^\top - (\kappa_0 + \tilde{n}_k)\mu_k\mu_k^\top}{\nu_0 + \tilde{n}_k - d - 1} \end{aligned} \tag{41}$$

where again $\tilde{n}_k = |\{i : z_i = k\}|$.

C.1 Gibbs Derivation

In the last section we used EM for inference and now we turn to Gibbs Sampling, another popular method. Gibbs sampling is a variety of MCMC sampling in which we cycle through all our latent random variables, resampling each conditioned on the currently sampled values of all other random variables.

Gibbs sampling is an MCMC method that traditionally sweeps all the variables in each iteration and one at a time, samples each variable conditioned on the rest (using $p(z_i \mid rest)$, the full conditional). We can often do better (consequence of the Rao-Blackwell theorem) by collapsing and integrating out θ_k and π. See Algorithm 4, and see Appendix C for more details.

The Markov blanket for z_i becomes $x_{-i}, z_{-i}, x_i, \phi_0, m_0$, and α in this case. We obtain

$$\begin{aligned} P(z_i \mid rest) &= \frac{P(x_i \mid z_i, \{x_j : z_j = z_i\}, m_0, \phi_0) P(z_i \mid z_{-i}, \alpha)}{P(x_i \mid x_{-i}, m_0, \phi_0)} \\ &= \frac{\exp[h(m_{z_i}, \phi_{z_i}) - h(m_{z_i} - 1, \phi_{z_i} - \phi(x_i))](n_{z_i} - 1)}{\sum_{j=1}^{K} \exp[h(m_j, \phi_j) - h(m_j - 1, \phi_j - \phi(x_i))](n_j - 1)} \end{aligned} \tag{42}$$

Note that $m_k - m_0 = n_k - \alpha_k$. Lset $\tilde{n}_k = |\{i : z_i = k\}|$ so that $m_k = m_0 + \tilde{n}_k$ and $n_k = \tilde{n}_k + \alpha$. Thus, we need to maintain only two invariants, n_k and ϕ_k per component in the inference procedure.

Algorithm 4 Collapsed Gibbs sampling for mixture models

Initialize $\boldsymbol{z}$ randomly and evaluate initial counts $\tilde{n}_k$ and statistics ϕ_k.
$t \leftarrow 0$
while $t \leq T$ **do**
  for $i = 1 \rightarrow N$ **do**
    Remove datum from current component and update statistics: $\tilde{n}_{z_i} \leftarrow \tilde{n}_{z_i} - 1, \phi_{z_i} \leftarrow \phi_{z_i} - \phi(x_i)$
    Sample z_i using the PMF stored in $p[k] \leftarrow (\alpha + \tilde{n}_k - 1) \exp\left(h(m_0 + \tilde{n}_k + 1, \phi_k + \phi(x_i)) - h(m_0 + \tilde{n}_k, \phi_k)\right)$; $p \leftarrow p/sum(p)$;
    Add datum to the new component and update statistics: $\tilde{n}_{z_i} \leftarrow \tilde{n}_{z_i} + 1, \phi_{z_i} \leftarrow \phi_{z_i} + \phi(x_i)$
  end for
  $t \leftarrow t + 1$
end while

D (S)EM Derivation for LDA

We derive an EM procedure for LDA.

D.1 LDA Model

In LDA, we model each document m of a corpus of M documents as a distribution θ_m that represents a mixture of topics. There are K such topics, and we model each topic k as a distribution ϕ_k over the vocabulary of words that appear in our corpus. Each document m contains N_m words w_{mn} from a vocabulary of size V, and we associate a latent variable z_{mn} to each of the words. The latent variables can take one of K values that indicate which topic the word belongs to. We give each of the distributions θ_m and ϕ_k a Dirichlet prior, parameterized respectively with a constant α and β. More concisely, LDA has the following mixed density.

$$p(\boldsymbol{w}, \boldsymbol{z}, \boldsymbol{\theta}, \boldsymbol{\phi}) = \left[\prod_{m=1}^{M}\prod_{n=1}^{N_m} \mathrm{Cat}(w_{mn} \mid \phi_{z_{mn}})\ \mathrm{Cat}(z_{mn} \mid \theta_m)\right]\left[\prod_{m=1}^{M} \mathrm{Dir}(\theta_m \mid \alpha)\right]\left[\prod_{k=1}^{K} \mathrm{Dir}(\phi_k \mid \beta)\right] \tag{43}$$

The choice of a Dirichlet prior is not a coincidence: we can integrate all of the variables θ_m and ϕ_k and obtain the following closed form solution.

$$p(\boldsymbol{w}, \boldsymbol{z}) = \left[\prod_{m=1}^{M} \mathrm{Pol}\big(\{z_{m'n} \mid m' = m\}, K, \alpha\big)\right]\left[\prod_{k=1}^{K} \mathrm{Pol}\big(\{w_{mn} \mid z_{mn} = k\}, V, \beta\big)\right] \tag{44}$$

where Pol is the Polya distribution

$$\mathrm{Pol}(S, X, \eta) = \frac{\Gamma(\eta\, K)}{\Gamma(|S| + \eta\, X)} \prod_{x=1}^{X} \frac{\Gamma\big(\big|\{z \mid z \in S, z = x\}\big| + \eta\big)}{\Gamma(\eta)} \tag{45}$$

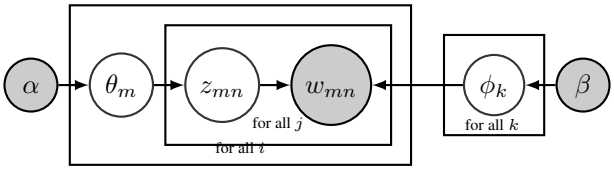


Figure 4: LDA Graphical Model

Algorithm 5 LDA Generative Model

input: $\boldsymbol{\alpha}, \boldsymbol{\beta}$

1: **for** $k = 1 \to K$ **do**
2: Choose topic $\boldsymbol{\phi}_k \sim \mathsf{Dir}(\boldsymbol{\beta})$
3: **end for**
4: **for all** document m in corpus D **do**
5: Choose a topic distribution $\boldsymbol{\theta}_m \sim \mathsf{Dir}(\boldsymbol{\alpha})$
6: **for all** word index n from 1 to N_m **do**
7: Choose a topic $z_{mn} \sim \mathsf{Categorical}(\boldsymbol{\theta}_m)$
8: Choose word $w_{mn} \sim \mathsf{Categorical}(\boldsymbol{\phi}_{z_{mn}})$
9: **end for**
10: **end for**

The joint probability density can be expressed as:

$$\begin{aligned} p(W, Z, \theta, \phi|\alpha, \beta) &= \left[\prod_{k=1}^{K} p(\phi_k|\beta)\right]\left[\prod_{m=1}^{M} p(\theta_m|\alpha) \prod_{n=1}^{N_m} p(z_{mn}|\theta_m) p(w_{mn}|\phi_{z_{mn}})\right] \\ &\propto \left[\prod_{k=1}^{K}\prod_{v=1}^{V} \phi_{kv}^{\beta-1}\right]\left[\prod_{m=1}^{M}\left(\prod_{k=1}^{K} \theta_{mk}^{\alpha-1}\right)\prod_{n=1}^{N_m} \theta_{mz_{mn}} \phi_{z_{mn}w_{mn}}\right] \end{aligned} \tag{46}$$

D.2 Expectation Maximization

We begin by marginalizing the latent variable Z and finding the lower bound for the likelihood/posterior:

$$\begin{aligned}
\log p(W, \theta, \phi|\alpha, \beta) &= \log \sum_Z p(W, Z, \theta, \phi|\alpha, \beta) \\
&= \sum_{m=1}^{M} \sum_{n=1}^{N_m} \log \sum_{k=1}^{K} p(z_{mn} = k|\theta_m) p(w_{mn}|\phi_k) \\
&\qquad + \sum_{k=1}^{K} \log p(\phi_k|\beta) + \sum_{m=1}^{M} \log p(\theta_m|\alpha) \\
&= \sum_{m=1}^{M} \sum_{n=1}^{N_m} \log \sum_{k=1}^{K} q(z_{mn} = k|w_{mn}) \frac{p(z_{mn} = k|\theta_m) p(w_{mn}|\phi_k)}{q(z_{mn} = k|w_{mn})} \\
&\qquad + \sum_{k=1}^{K} \log p(\phi_k|\beta) + \sum_{m=1}^{M} \log p(\theta_m|\alpha) \\
\text{(Jensen Inequality)} \qquad &\geq \sum_{m=1}^{M} \sum_{n=1}^{N_m} \sum_{k=1}^{K} q(z_{mn} = k|w_{mn}) \log \frac{p(z_{mn} = k|\theta_m) p(w_{mn}|\phi_k)}{q(z_{mn} = k|w_{mn})} \\
&\qquad + \sum_{k=1}^{K} \log p(\phi_k|\beta) + \sum_{m=1}^{M} \log p(\theta_m|\alpha)
\end{aligned} \tag{47}$$

Let us define the following functional:

$$\begin{aligned}
F(q, \theta, \phi) := &- \sum_{m=1}^{M} \sum_{n=1}^{N_m} D_{KL}(q(z_{mn}|w_{mn})||p(z_{mn}|w_{mn}, \theta_m, \phi)) \\
&+ \sum_{m=1}^{M} \sum_{n=1}^{N_m} p(w_{mn}|\theta_m, \phi) + \sum_{k=1}^{K} \log p(\phi_k|\beta) + \sum_{m=1}^{M} \log p(\theta_m|\alpha)
\end{aligned} \tag{48}$$

D.2.1 E-Step

In the E-step, we fix θ, ϕ and maximize F for q. As q appears only in the KL-divergence term, it is equivalent to minimizing the KL-divergence between $q(z_{mn}|w_{mn})$ and $p(z_{mn}|w_{mn}, \theta_m, \phi)$. We know that for any distributions f and g the KL-divergence is minimized when $f = g$ and is equal to 0. Thus, we have

$$\begin{aligned}
q(z_{mn} = k|w_{mn}) &= p(z_{mn} = k|w_{mn}, \theta_m, \phi) \\
&= \frac{\theta_{mk} \phi_{k w_{mn}}}{\sum_{k'=1}^{K} \theta_{mk'} \phi_{k' w_{mn}}}
\end{aligned} \tag{49}$$

For simplicity of notation, let us define

$$\boxed{q_{mnk} = \frac{\theta_{mk} \phi_{k w_{mn}}}{\sum_{k'=1}^{K} \theta_{mk'} \phi_{k' w_{mn}}}} \tag{50}$$

D.2.2 M-Step

In the E-step, we fix q and maximize F for θ, ϕ. As this will be a constrained optimization (θ and ϕ must lie on simplex), we use standard constrained optimization procedure of Lagrange multipliers.

The Lagrangian can be expressed as:

$$\begin{aligned}\mathcal{L}(\theta,\phi,\lambda,\mu) &= \sum_{m=1}^{M}\sum_{m=1}^{N_m}\sum_{k=1}^{K} q(z_{mn}=k|w_{mn})\log\frac{p(z_{mn}=k|\theta_m)p(w_{mn}|\phi_k)}{q(z_{mn}=k|w_{mn})} + \sum_{k=1}^{K}\log p(\phi_k|\beta) \\ &\quad + \sum_{m=1}^{M}\log p(\theta_m|\alpha) + \sum_{k=1}^{K}\lambda_k\left(1-\sum_{v=1}^{V}\phi_{kv}\right) + \sum_{m=1}^{M}\mu_i\left(1-\sum_{k=1}^{K}\theta_{mk}\right) \\ &= \sum_{m=1}^{M}\sum_{n=1}^{N_m}\sum_{k=1}^{K} q_{mnk}\log\theta_{mk}\phi_{kw_{mn}} + \sum_{k=1}^{K}\sum_{v=1}^{V}(\beta_v-1)\log\phi_{kv} + \sum_{m=1}^{M}\sum_{k=1}^{K}(\alpha_k-1)\log\theta_{mk} \\ &\quad + \sum_{k=1}^{K}\lambda_k\left(1-\sum_{v=1}^{V}\phi_{kv}\right) + \sum_{m=1}^{M}\mu_m\left(1-\sum_{k=1}^{K}\theta_{mk}\right) + \text{const.}\end{aligned} \tag{51}$$

Maximizing θ Taking derivative with respect to θ_{mk} and setting it to 0, we obtain

$$\begin{aligned}\frac{\partial\mathcal{L}}{\partial\theta_{mk}} = 0 &= \sum_{j=1}^{N_m}\frac{q_{mnk}+\alpha_k-1}{\theta_{mk}} - \mu_m \\ \mu_m\theta_{mk} &= \sum_{j=1}^{N_i} q_{mnk} + \alpha_k - 1\end{aligned} \tag{52}$$

After solving for μ_m, we finally obtain

$$\theta_{mk} = \frac{\sum_{n=1}^{N_m} q_{mnk}+\alpha_k-1}{\sum_{k'=1}^{K}\sum_{j=1}^{N_m} q_{mnk'}+\alpha_{k'}-1} \tag{53}$$

Note that $\sum_{k'=1}^{K} q_{mnk'} = 1$, we reach at the optimizer:

$$\boxed{\theta_{mk} = \frac{1}{N_m + \sum(\alpha_{k'}-1)}\left(\sum_{n=1}^{N_m} q_{mnk}+\alpha_k-1\right)} \tag{54}$$

Maximizing ϕ Taking derivative with respect to ϕ_{kv} and setting it to 0, we obtain

$$\begin{aligned}\frac{\partial\mathcal{L}}{\partial\phi_{kv}} = 0 &= \sum_{m=1}^{M}\sum_{n=1}^{N_m}\frac{q_{mnk}\delta(v-w_{mn})+\beta_v-1}{\phi_{kv}} - \lambda_k \\ \lambda_k\phi_{kv} &= \sum_{m=1}^{M}\sum_{n=1}^{N_m} q_{mnk}\delta(v-w_{mn}) + \beta_v - 1\end{aligned} \tag{55}$$

After solving for λ_k, we finally obtain

$$\phi_{kv} = \frac{\sum_{m=1}^{M}\sum_{n=1}^{N_m} q_{mnk}\delta(v-w_{mn})+\beta_v-1}{\sum_{v'=1}^{V}\sum_{m=1}^{M}\sum_{n=1}^{N_m}\delta(v'-w_{mn})+\beta_{v'}-1} \tag{56}$$

Note that $\sum_{v'=1}^{V}\delta(v'-w_{mn}) = 1$, we reach at the optimizer:

$$\boxed{\phi_{kv} = \frac{\sum_{m=1}^{M}\sum_{n=1}^{N_m} q_{mnk}\delta(v-w_{mn})+\beta_v-1}{\sum_{m=1}^{M}\sum_{n=1}^{N_m} q_{mnk}+\sum(\beta_{v'}-1)}} \tag{57}$$

D.3 Introducing Stochasticity

After performing the E-step, we add an extra simulation step, i.e. we draw and impute the values for the latent variables from its distribution conditioned on data and current estimate of the parameters. This means basically q_mnk gets transformed into $\delta(z_{mn}-\tilde{k})$ where $\tilde{k}$ is value drawn from the conditional distribution. Then we proceed to perform the M-step, which is even simpler now. To summarize SEM for LDA will have following steps:

E-step in parallel compute the conditional distribution locally:

$$q_{mnk} = \frac{\theta_{mk}\phi_{kw_{mn}}}{\sum_{k'=1}^{K}\theta_{mk'}\phi_{k'w_{ij}}} \tag{58}$$

S-step in parallel draw z_{mn} from the categorical distribution:

$$z_{mn} \sim \mathsf{Categorical}(q_{mn1}, ..., q_{mnK}) \tag{59}$$

M-step in parallel compute the new parameter estimates:

$$\begin{aligned} \theta_{mk} &= \frac{D_{mk} + \alpha_k - 1}{N_m + \sum(\alpha_{k'} - 1)} \\ \phi_{kv} &= \frac{W_{kv} + \beta_v - 1}{T_k + \sum(\beta_{v'} - 1)} \end{aligned} \tag{60}$$

where $D_{mk} = \Big|\big\{\, z_{mn} \mid z_{mn} = k \,\big\}\Big|$,
$W_{kv} = \Big|\big\{\, z_{mn} \mid w_{mn} = v,\, z_{mn} = k \,\big\}\Big|$, and
$T_k = \Big|\big\{\, z_{mn} \mid z_{mn} = k \,\big\}\Big| = \sum_{v=1}^{V} W_{kv}$.