Latent Dirichlet Allocation

anhminhphanle@gmail.com

February 2018

The purpose of this summary is to 1/ understand main components that build up LDA, the intuition how LDA works, 2/ the maths' derivation the author didn't include in the original paper and 3/the author's code implementation. These three are equally important in order to understand LDA theorically and practically. Let's start to look at the Dirichlet distribution.

1 Dirichlet Distribution

To understand Dirichlet distribution, let's start with a simpler case where $K = 2$, the Dirichlet distribution becomes Beta distribution. Let $X = [X_1, X_2]$ is the random pmf of $\text{Beta}(\alpha, \beta)$. $X \sim \text{Beta}(\alpha, \beta)$ when:

$$\begin{aligned} f(x;\alpha,\beta) &= \text{Beta}(\alpha,\beta) \\ &= \frac{1}{\mathrm{B}(\alpha,\beta)} x^{\alpha-1}(1-x)^{\beta-1} \\ &= \frac{\Gamma\big(\alpha+\beta\big)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1}, \end{aligned} \tag{1.1}$$

You can see that the Bionomial distribution has a similar form $p^k(1-p)^{n-k}$:

> "the binomial distribution with parameters n and p is the discrete probability distribution of **the number of successes in a sequence** of n independent experiments, each asking a yes–no question" - Binomial distribution, Wikipedia

With a fixed $x_i = p_i$, $\alpha - 1 = k$, $\beta - 1 = n - k$, we have $\alpha + \beta = n - 2$ and $x^{\alpha-1}(1-x)^{\beta-1} \sim p^k(1-p)^{n-k}$. Now take an example of throwing a coin of two outcomes head $\{H\}$ or tail $\{T\}$ "in sequence or continously". Let a "biased" coin with $\alpha = 100, \beta = 200$, i.e. $X = [X_1, X_2] = [100\ H\text{s}, 200\ T\text{s}]$. The probability of a 100 successful outcomes of H in a 300 sequences of throwing coin with a random probability x (of only a single success H) is

Beta(α, β). Now, consider its means:

$$\begin{aligned} \mu = \mathrm{E}[X_1] &= \int_0^1 x f(x;\alpha,\beta) dx \\ &= \int_0^1 \frac{x^{\alpha-1}(1-x)^{\beta-1}}{\mathrm{B}(\alpha,\beta)} \\ &= \frac{\alpha}{\alpha+\beta}, \end{aligned} \tag{1.2}$$

Also the median $\approx \frac{\alpha-\frac{1}{3}}{\alpha+\beta-\frac{2}{3}}$, for $\alpha, \beta \geq 1$. With whatever the shape of this distribution is, its mean and median should be those numbers. The Beta distribution is simply defined with $\alpha + \beta$ experiments, with random variable of a single success outcome probability x, the expectation $\mathrm{E}[X_1 = \alpha$ times of $x] = \frac{\alpha}{\alpha+\beta}$. We can re-state our definition as:

> The Beta distribution with parameter α and β is the discrete probability distribution of α times successful outcomes with a random variable x in a sequence of $\alpha + \beta$ independent experiments with a probability of $\frac{\alpha}{\alpha+\beta}$.

Note that we put the statement of expected value $\frac{\alpha}{\alpha+\beta}$ in order to differentiate from the bionomial distribution. Let's try this on R software:

Listing 1: Beta Distribution Illustration

```r
# Generate random x, 0\leq x\leq 1 to see whether the mean of distribution moves
x <- runif(1, min = 0, max = 1)
# sample 100 times with \alpha = 100, \beta = 200
beta <- rbeta(100,100,200)
mean <- mean(beta) # mean = 0.3312285
plot(beta)
abline(h=mean,col='red')
curve(dbeta(x,100,200))
abline(v=mean,col='red')
```

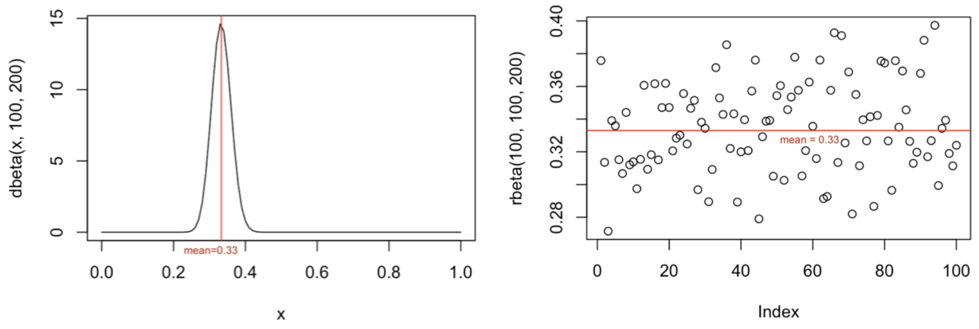

Figure 1: 1/Beta Distribution $0 \leq x \leq 1, (\alpha, \beta) = (100, 200)$ and 2/Sampling from Beta distribution

Now if your friend come to you and tell that he's writting a game for people to roll a 6-sides dice. He has observed that the set of sides $\{i\}_{i=1}^{K=6}$, such that side ith appears α_i times in every sequence of $\sum_{j=1}^{6} \alpha_i$ with the probability of $\alpha_i / \sum_{j=1}^{6} \alpha_i$. This has been drawing a large number of people to play. Now he needs you to develop an algorithm to build such a "loaded" dice. How can you solve this? You have a set of outcomes of rolling a 6-sides dice $X = [X_1, X_2, X_3, X_4, X_5, X_6]$ and you know the prior mean value of each outcome $\mathrm{E}[X_i] = \alpha_i / \sum_{j=1}^{6} \alpha_i$. You wish that if the dice only has 2 sides so you can use Beta distribution to solve your problem right away. Fortunately, this is where the Dirichlet distribution comes to the rescue. It's the expansion form of Beta distribution for $K > 2$.

With a set of outcomes $X = [X_1, X_2, \dots, X_K]$ be the random pmf, $X \sim \mathrm{Dir}(\alpha)$ when:

$$\begin{aligned} f(\boldsymbol{x}, \boldsymbol{\alpha}) &= \frac{1}{\mathrm{B}(\boldsymbol{\alpha})} \prod_{i=1}^{K} x_i^{\alpha_i - 1} \\ &= \frac{\Gamma\left(\sum_{i=1}^{K} \alpha_i\right)}{\prod_{i=1}^{K} \Gamma(\alpha_i)} \prod_{i=1}^{K} x_i^{\alpha_i - 1}, \end{aligned} \tag{1.3}$$

where $\Gamma(.)$ is the gamma function, for positive integer n, $\Gamma(n) = (n-1)!$. We can also re-define the definition of Dirichlet distribution:

> The Dirichlet distribution with parameters $\{\alpha_i\}_{i=1}^{K}$ is the discrete probability distribution of α_i times successful outcomes with a random variable x_i in a sequence of $\sum_{j=1}^{K} \alpha_j$ independent experiments with a probability of $\alpha_i / \sum_{j=1}^{K} \alpha_j$.

The Dirichlet distribution also looks something similar to multinomial distribution, but instead of in the form $\prod_{i=1}^{K} p_i^{x_i - 1}$ it takes $\prod_{i=1}^{K} x_i^{\alpha_i - 1}$. You can use R to sample from Dirichlet distribution:

Listing 2: Sample from Dirichlet distribution with K

```
install.packages('MCMCpack') #Skip if you already installed
library(MCMCpack)
alpha <- runif(6, min = 0, max = 100) # K = 6
a <- as.data.frame(rdirichlet(100, alpha))
alpha/sum(alpha) # prior mean
# Now check if posterior mean equal to prior mean
mean(a$V1)
mean(a$V2)
```

From our new definition, if we have a corpus D with K topics, to find a distribution of α_i times of topic ith appears with a probability p_i such that all topics' probabilities $\sum_{i=1}^{K} p_i = 1$, we can use Dirichlet distribution to build up a LDA model.

The last thing we might need to consider another aspect of the distribution; its variance:

$$\text{Var}[X] = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$$

Intuitively $\text{Var}[X] \approx \frac{1}{8\alpha}$ if $\alpha = \beta$. In general $\text{Var}[X] \not\propto (\alpha, \beta)$. Let's try some values of (α, β) on R to see how the shape changes:

Listing 3: Beta Distribution Illustration

```
x <- runif(1, min = 0, max = 1)
curve(dbeta(x,0.1,0.1))
curve(dbeta(x,10,10))
curve(dbeta(x,100,100))
```

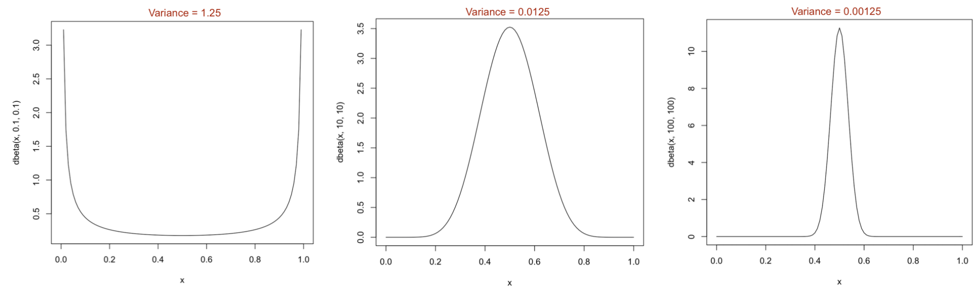

Figure 2: Illustrate the shape of Beta distribution

We can generalize to the Dirichlet distribution that very low $\boldsymbol{\alpha}$ leads to a very low sparse curve in middle, like a bowl, or a valley and high value at both two sides; this is due to the high variance of two vertical cliffs and the "flat valley". Similarly high $\boldsymbol{\alpha}$ leads to a high peak, due to small variance and all the samples tend to merge into one vertical line towards the mean. If you are reading this post because of LDA, these are the things most of Dirichlet distribution papers uses the simplex to illustrate the density of Dirichlet distribution without explaining the reason behind, this perhaps creating complexity for the reader. This writting of Dirichlet distribution helps for a better understanding of the intuition of Dirichlet distribution and reading LDA papers. Next, we use Dirichlet distribution to formulate LDA's model.

2 LDA's model

Let k be the latent number of topics, D be the number of documents and V be the number of terms in the vocabulary. Note that we use k not K because it's a variable we will sample from the Dirichlet distribution. We use α to draw a the distribution of k topics over documents d. And use β to draw a distribution of topics over words w_d. We use i to index a topic, d to index a document, j to index a word and w to denote a word. In LDA, $\alpha_{k\times 1}$ and $\beta_{k\times V}$ are model parameters, while $\theta_{D\times k}$ and $\boldsymbol{z}$ are hidden variables. Given these distributions, the LDA generative process is as follows:

1. For each documents:
   (a) Choose a distribution d_1 over topics, for example "70% topic A, 20% topic B, 10% topic C".
   (b) For each word in the document:
      i. Draw a topic i in one of the k topics with a distribution d_2 conditioned on d_1. For example, if we pick topic A, the document will have 70% chance of containing topic A.
      ii. Draw a word amongst many words in topic i with a distibution d_3 conditioned on d_2.

This model shows that:

- A document contains a mixture of topics. For example, document d contains "70% topic A, 20% topic B, 10% topic C".
- A topic contains a mixture of words. "Education" might contain words like "school", "teachers", "elementary".

After having a rough understanding, we formalize the above process as follows:

1. For each topic i in k
   (a) Draw a topic distribution over words $\phi_i \sim \text{Dir}(\beta)$.
2. For each word n in the document d:
   (a) Draw a topic distribution over documents, $\theta_d \sim \text{Dir}(\alpha)$
   (b) For each word in the document:
      i. Draw a specific topic $z_{d,n} \sim \text{multi}(\theta)$
      ii. Draw a word $w_{d,n} \sim \beta_{z_{d,n}}$,

where ϕ_i the probability over the vocabulary generated by latent topic i, $\text{Dir}(\alpha)$ is a draw from a uniform Dirichlet distribution with parameter α and multi(.) is a Multinomial distribution. Note that $w_{d,n} \sim p(w_{d,n}|z_{d,n}, \beta_i) = \beta_{z_{d,n},i} = \beta_{z_{d,n}}$, because z_n is already i we have picked from the beginning. The subsript n, d or d is just to express that we're only considering the word n and document d.

Why Dirichlet? The LDA generative process has 3-levels of distribution. The distribution d_1 is actually a Dirichlet distribution. Similarly to the multinomial distribution, from the pmf itself $\frac{n!}{\prod x_i} \prod p_i^{x_i}$, each topic with probability p_i can appear x_i times in all the corpus D. And the author also stated that "The Dirichlet is a convenient distribution on the simplex — it is in the **exponential family**, has **finite dimensional** sufficient statistics, and is **conjugate to the multinomial distribution**".

> "For example, it models the probability of counts for rolling a k-sided dice n times. For n independent trials each of which leads to a success for exactly one of k categories, with each category having a given fixed success probability, the multinomial distribution gives the probability of any particular combination of numbers of successes for the various categories." - Multinomial distribution, Wikipedia.

where n is the number of times all topics appear in a corpus, k categorical is the number of topics. The multinomial distribution can express partially the distribution of topics in a corpus, and Dirichlet is conjugate to it.

What roles α, β play? Consider if 3 topics have the same probability, we choose $\alpha(10, 10, 10)$, if we want the first topic to appear more, we choose $\alpha(30, 10, 10)$. We can see how α represents the topic mixtures. We then draw a sample from this Dir(α) to get a vector θ that sums up to 1. This θ will be the weights of a k-sided dice. If we throw the dice and get 3, then we will generate a word in 3th topic. To this point, we're able to understand the process from $\alpha \rightarrow \theta \rightarrow z_{i=3}$. Similarly, we draw ϕ_i from Dir(β) and then throw V sides dices with weight ϕ to get a 7th word in V $w_{j=7}$. We infer a similar process $\beta \rightarrow \phi \rightarrow w_{j=7}$. Note that we sample ϕ from β using (3.10) that will be derived later. This is the reason the author didn't mention what to sample from β in the beginning of the paper, and also it's not included in the graphical model in Fig.3, but only a direct connection from $\beta \rightarrow w$.

Next, to solve the problem of LDA we need to compute:

$$p(\theta, \boldsymbol{z}|\boldsymbol{w}, \alpha, \beta) = \frac{p(\theta, \boldsymbol{z}, \boldsymbol{w}|\alpha, \beta)}{p(\boldsymbol{w}|\alpha, \beta)}. \tag{2.1}$$

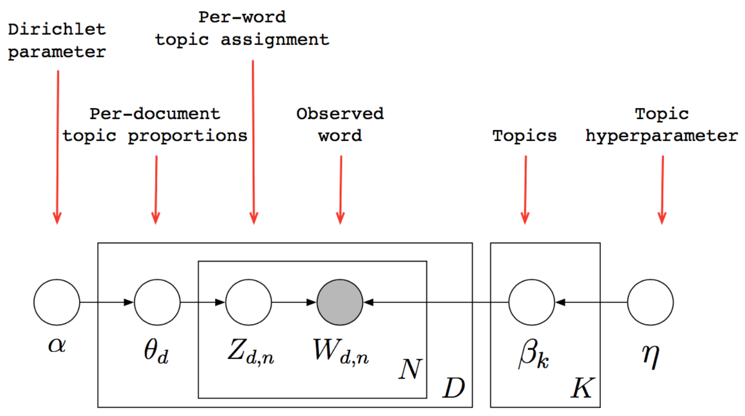

Figure 3: LDA graphical model

However, this distribution is intractable to compute. That's where Variational Inference comes about. The idea of Variational Inference is to find and optimize a closest family to the posterior by making use of Jensen's inequality to obtain an adjustable lower bound on the log likelihood. Because we are only interested in a family (γ, ϕ) closest to $(\theta, \boldsymbol{z})$, we can drop some edges and nodes of Fig.3 to simplify (2.1) as below:

$$q(\theta, \boldsymbol{z}|\gamma, \phi) = q(\theta|\gamma) \prod_{n=1}^{N} q(z_n|\phi_n) \tag{2.2}$$

The main idea is that we use variational expectation-maximization (EM):

1. (E-step) For each document, find the optimizing values of the variational parameters $\{\gamma_d^*, \phi_d^* : d \in D\}$.
2. (M-step) Maximize the resulting lower bound on the log likelihood with respect to the model parameters α and β. This corresponds to finding maximum likelihood estimates with expected sufficient statistics for each document under the approximate posterior which is computed in the E-step.

3 E-step: optimizing values of the variational parameters γ_d, ϕ_d

3.1 Find a function L for the expectation of the log likelihood

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

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

Using Jensen's inequality we have:

$$\begin{aligned}
\log p(\boldsymbol{w}|\alpha, \beta) &= \log \int \sum_{\boldsymbol{z}} p(\theta, \boldsymbol{z}, \boldsymbol{w}|\alpha, \beta) d\theta \\
&= \log \int \sum_{\boldsymbol{z}} \frac{p(\theta, \boldsymbol{z}, \boldsymbol{w}|\alpha, \beta) q(\theta, \boldsymbol{z})}{q(\theta, \boldsymbol{z})} d\theta \\
&\geq \int \sum_{\boldsymbol{z}} q(\theta, \boldsymbol{z}) p(\theta, \boldsymbol{z}, \boldsymbol{w}|\alpha, \beta) q(\theta, \boldsymbol{z}) - \int \sum_{\boldsymbol{z}} q(\theta, \boldsymbol{z}) q(\theta, \boldsymbol{z}) \\
&= \mathrm{E}_q[\log p(\theta, \boldsymbol{z}, \boldsymbol{w}|\alpha, \beta)] - \mathrm{E}_q[\log q(\theta, \boldsymbol{z})]
\end{aligned} \tag{3.1}$$

The difference between LHS and RHS of Eq.(4.3) is the KL divergence between the variational $q(\theta, \boldsymbol{z}|\gamma, \phi)$ and true posterior probability $p(\theta, \boldsymbol{z}, \boldsymbol{w}|\alpha, \beta)$. Let RHS of Eq.(4.3) $\triangleq L(\gamma, \phi; \alpha, \beta)$ we have

$$\log p(\boldsymbol{w}|\alpha, \beta) = L(\gamma, \phi; \alpha, \beta) + D_{\mathrm{KL}}(q(\theta, \boldsymbol{z}|\gamma, \phi) \,||\, p(\theta, \boldsymbol{z}, \boldsymbol{w}|\alpha, \beta)) \tag{3.2}$$

This shows that maximize $L(\gamma,\phi;\alpha,\beta) \widehat{=}$ minimize D_{KL}. We need to find a optimal solutions where:

$$(\gamma^*,\phi^*) = \arg\min_{\gamma,\phi} \mathrm{D}(q(\theta,\mathbf{z}|\gamma,\phi)||p(\theta,\boldsymbol{z}|\boldsymbol{w},\alpha,\beta)). \tag{3.3}$$

3.2 Optimize L

Expand (3.2):

$$\begin{aligned} L(\gamma,\phi;\alpha,\beta) = {}& \mathrm{E}_q[\log p(\theta|\alpha)] + \mathrm{E}_q[\log p(\boldsymbol{z}|\theta)] + \mathrm{E}_q[\log p(\boldsymbol{w}|\boldsymbol{z},\beta)] \\ & - \mathrm{E}_q[\log p(\theta)] - \mathrm{E}_q[\log p(\boldsymbol{z})] \end{aligned} \tag{3.4}$$

Consider the first term:

$$\begin{aligned} & \mathrm{E}_q[\log p(\theta|\alpha)] \\ & = \mathrm{E}_q[\log\Big(\exp\Big\{\Big(\sum_{j=1}^{k}(\alpha_i-1)\log\theta_i\Big)\Big\} + \log\Gamma\Big(\sum_{i=1}^{k}\alpha_i\Big) - \sum_{i=1}^{k}\log\Gamma(\alpha_i)\Big\}\Big)] \\ & = \Big(\sum_{i=1}^{k}(\alpha_i-1)\mathrm{E}_q[\log\theta_i]\Big) + \log\Gamma\Big(\sum_{i=1}^{k}\alpha_i\Big) - \sum_{i=1}^{k}\log\Gamma(\alpha_i) \\ & = \Bigg(\sum_{i=1}^{k}(\alpha_i-1)\Big(\Psi(\alpha_i) - \Psi\Big(\sum_{j=1}^{k}\alpha_j\Big)\Big)\Bigg) + \log\Gamma\Big(\sum_{i=1}^{k}\alpha_i\Big) - \sum_{i=1}^{k}\log\Gamma(\alpha_i). \end{aligned}$$

Let z_n the topic assignment of the nth word in current document. $z_{n,i}=1$ when the topic assignment is the ith topic, otherwise $z_{n,i}=0$. $p(z_n|\theta)$ is simply θ_i when $z_{n,i}=1$. Let $p(z_{n,i}|\theta_i) = \theta_i^{z_{n,i}}$ for the unique i so that:

$$\begin{aligned} \log p(z_{n,i}|\theta_i) &= \log\theta_i^{z_{n,i}} \\ &= z_{n,i}\log\theta_i. \end{aligned} \tag{3.5}$$

Expand the second term:

$$\begin{aligned}
\mathrm{E}_q[\log p(\boldsymbol{z}|\theta)] &= \sum_{n=1}^{N} \mathrm{E}_q[\log p(\boldsymbol{z}_n|\theta)] \\
&= \sum_{n=1}^{N} \sum_{i=1}^{k} \mathrm{E}_q[\log p(\boldsymbol{z}_{n,i}|\theta_i)] \\
&= \sum_{n=1}^{N} \sum_{i=1}^{k} \mathrm{E}_q[\log \theta_i^{z_{n,i}}] \\
&= \sum_{n=1}^{N} \sum_{i=1}^{k} \mathrm{E}_q[z_{n,i} \log \theta_i] \\
&= \sum_{n=1}^{N} \sum_{i=1}^{k} \mathrm{E}_q[z_{n,i}] \mathrm{E}_q[\log \theta_i] \\
&= \sum_{n=1}^{N} \sum_{i=1}^{k} \phi_{ni} \Big(\Psi(\gamma_i) - \Psi\Big(\sum_{j=1}^{k} \gamma_i \Big) \Big).
\end{aligned}$$

Let $\beta_{ij} = p(w_{n,j} = 1|z_i = 1)$, recall that each w_n is a vector of size V with exactly one component equal to one; we can select the unique j such that $w_{n,j} = 1$. Constraint similiarly to (3.5), we have:

$$\begin{aligned}
\log p(\boldsymbol{w}|\boldsymbol{z}, \beta) &= \log \beta_{ij}^{z_{n,i}.w_{n,j}} \\
&= z_{n,i}.w_{n,j} \log \beta_{ij}.
\end{aligned} \tag{3.6}$$

Third term:

$$\begin{aligned}
\mathrm{E}_q[\log p(\boldsymbol{w}|\boldsymbol{z}, \beta) &= \sum_{n=1}^{N} \mathrm{E}_q[\log p(w_n|z_n, \beta)] \\
&= \sum_{n=1}^{N} \sum_{i=1}^{k} \sum_{j=1}^{V} \mathrm{E}_q[\log p(w_{n,j}|z_{n,i}, \beta_{ij})] \\
&= \sum_{n=1}^{N} \sum_{i=1}^{k} \sum_{j=1}^{V} \mathrm{E}_q[\log \beta_{ij}^{z_{n,i}.w_{n,j}}] \\
&= \sum_{n=1}^{N} \sum_{i=1}^{k} \sum_{j=1}^{V} \mathrm{E}_q[z_{n,i}] \mathrm{E}_q[w_{n,j}] \mathrm{E}_q[\log \beta_{ij}] \\
&= \sum_{n=1}^{N} \sum_{i=1}^{k} \sum_{j=1}^{V} \phi_{n,i} w_{n,j} \log \beta_{ij}.
\end{aligned}$$

Expand the fourth term similarly the first term (just change α to γ) and fifth terms, we have:

$$
\begin{aligned}
&-\mathrm{E}_q[\log p(\theta)] - \mathrm{E}_q[\log p(\boldsymbol{z})] \\
&= \mathrm{E}_q[\log p(\theta|\gamma)] - \mathrm{E}_q[\log p(\boldsymbol{z}|\phi)] \\
&= -\Bigg(\sum_{i=1}^{k}(\gamma_i - 1)\Big(\Psi(\gamma) - \Psi\Big(\sum_{j=1}^{k}\gamma_j\Big)\Big)\Bigg) + \log\Gamma\Big(\sum_{i=1}^{k}\gamma_i\Big) - \sum_{i=1}^{k}\log\Gamma(\gamma_i) \\
&-\sum_{n=1}^{N}\sum_{i=1}^{k}\phi_{ni}\log\phi_{ni}.
\end{aligned}
$$

Sum up all of those five terms, we have the lower bound:

$$
\begin{aligned}
L(\gamma,\phi;\alpha,\beta) = \log\Gamma\Big(\sum_{j=1}^{k}\alpha_i\Big) &- \sum_{j=1}^{k}\log\Gamma(\alpha_i) + \sum_{j=1}^{k}(\alpha_i - 1)\Big(\Psi(\gamma_i) - \Psi\Big(\sum_{j=1}^{k}\gamma_i\Big)\Big) \\
&+ \sum_{n=1}^{N}\sum_{i=1}^{k}\phi_{ni}\Big(\Psi(\gamma_i) - \Psi\Big(\sum_{j=1}^{k}\gamma_i\Big)\Big) \\
&+ \sum_{n=1}^{N}\sum_{i=1}^{k}\sum_{j=1}^{V}\phi_{ni}w_{nj}\log\beta_{ij} \\
&- \log\Gamma\Big(\sum_{j=1}^{k}\gamma_i\Big) + \sum_{i=1}^{k}\log\Gamma(\gamma_i) - \sum_{i=1}^{k}(\gamma_i - 1)\Big(\Psi(\gamma_i) - \Psi\Big(\sum_{j=1}^{k}\gamma_i\Big)\Big) \\
&- \sum_{n=1}^{N}\sum_{i=1}^{k}\phi_{ni}\log\phi_{ni}.
\end{aligned}
\tag{3.7}
$$

We then maximize the lower bound $L(\gamma,\phi;\alpha,\beta)$ w.r.t ϕ and γ.

3.3 Maximize L w.r.t ϕ

ϕ_{ni} is the probability the nth word is generated by latent topic i, observed that $\sum_{i=1}^{k}\phi_{ni} = 1$. We only keep terms in L that have ϕ_{ni} and ignore those with only α_i or γ_i. Using Lagrange multiplier, we can rewrite Eq.(8.4):

$$
L_{[\phi_{ni}]} = \phi_{ni}\Big(\Psi(\gamma_i) - \Psi\Big(\sum_{j=1}^{k}\gamma_i\Big)\Big) + \phi_{ni}\log\beta_{ij} - \phi_{ni}\log\phi_{ni} + \underbrace{\lambda\Big(\sum_{j=1}^{k}\phi_{ni} - 1\Big)}_{\text{Lagrange term}}
$$

Taking derivative of L w.r.t ϕ_{ni}:

$$
\frac{\partial L}{\partial\phi_{ni}} = \Psi(\gamma_i) - \Psi\Big(\sum_{j=1}^{k}\gamma_i\Big) + \log\beta_{ij} - \log\phi_{ni} - 1 + \lambda
$$

Set this derivative to zero, we have maximum value of ϕ_{ni}:

$$\phi_{ni} \propto \beta_{iw_n} \exp\Big(\Psi(\gamma_i) - \Psi\Big(\sum_{j=1}^{k} \gamma_i\Big)\Big) \tag{3.8}$$

3.4 Maximize L w.r.t γ

Keep terms in Eq.(8.4) that contain γ_i:

$$\begin{aligned} L_{[\gamma]} &= \sum_{i=1}^{k} (\alpha_i - 1)\Big(\Psi(\gamma_i) - \Psi\Big(\sum_{j=1}^{k} \gamma_i\Big)\Big) + \sum_{n=1}^{N} \phi_{ni}\Big(\Psi(\gamma_i) - \Psi\Big(\sum_{j=1}^{k} \gamma_i\Big)\Big) \\ &\quad - \log\Gamma\Big(\sum_{j=1}^{k} \gamma_i\Big) + \log\Gamma(\gamma_i) - \sum_{i=1}^{k} (\gamma_i - 1)\Big(\Psi(\gamma_i) - \Psi\Big(\sum_{j=1}^{k} \gamma_i\Big)\Big) \\ &= \sum_{i=1}^{k} \Big(\Psi(\gamma_i) - \Psi\Big(\sum_{j=1}^{k} \gamma_i\Big)\Big)\Big(\alpha_i + \sum_{n=1}^{N} \phi_{ni} - \gamma_i\Big) - \log\Gamma\Big(\sum_{j=1}^{k} \gamma_i\Big) + \log\Gamma(\gamma_i) \end{aligned}$$

Taking derivative of $L_{[\gamma]}$ w.r.t γ_i:

$$\frac{\partial L}{\partial \gamma_i} = \Psi'(\gamma_i)\Big(\alpha_i + \sum_{n=1}^{N} \phi_{ni} - \gamma_i\Big) - \Psi'\Big(\sum_{j=1}^{k} \gamma_i\Big) \sum_{j=1}^{k} \Big(\alpha_i + \sum_{n=1}^{N} \phi_{ni} - \gamma_i\Big)$$

$L_{[\gamma]}$ is maximum when $\frac{\partial L}{\partial \gamma_i} = 0$ at:

$$\gamma_i = \alpha_i + \sum_{n=1}^{N} \phi_{ni} \tag{3.9}$$

3.5 Conlusion of E-step

The optimal γ^*, ϕ^* is found:

$$\phi_{ni} \propto \beta_{iw_n} \exp\{\mathrm{E}_q[\log\theta_i | \gamma]\} \tag{3.10}$$

$$\gamma_i = \alpha_i + \sum_{n=1}^{N} \phi_{ni} \tag{3.11}$$

Where

$$\mathrm{E}_q[\log\theta_i | \gamma] = \Psi(\gamma_i) - \Psi\Big(\sum_{j=1}^{k} \gamma_j\Big)$$

with $\Psi = \frac{d}{dx}\log(\Gamma(x))$, the digamma function.

4 M-step: maximizing the lower bound with respect to parameters α, β

With the set of documents D $\{\boldsymbol{w}_d\}_{d=1}^{M}$ we wish to find parameters α and β that maximize the log likelihood:

$$\ell(\alpha, \beta) = \sum_{d=1}^{M} \log p(\boldsymbol{w}_d | \alpha, \beta) \tag{4.1}$$

To maximize with respect to β, we isolate terms and add Lagrange multipliers:

$$L_{[\beta]} = \sum_{d=1}^{M} \sum_{n=1}^{N_d} \sum_{i=1}^{k} \sum_{j=1}^{V} \phi_{dni} w_{dnj} \log \beta_{ij} + \sum_{i=1}^{k} \lambda_i \Big(\sum_{j=1}^{V} \beta_{ij} - 1 \Big).$$

Take derivative w.r.t β_{ij}, set it to zero, and find:

$$\beta_{ij} \propto \sum_{d=1}^{M} \sum_{n=1}^{N_d} \phi_{dni} w_{dnj}.$$

The terms which contain α are:

$$L_{[\alpha]} = \sum_{d=1}^{M} \left(\log\Gamma(\sum_{j=1}^{k} \alpha_j) - \sum_{i=1}^{k} \log\Gamma(\alpha_i) + \sum_{i=1}^{k} \Big((\alpha_i - 1) \Big(\Psi(\gamma_{di}) - \Psi\Big(\sum_{j=1}^{k} \gamma_{dj} \Big)\Big)\Big) \right)$$

Taking the derivative w.r.t α_i:

$$\frac{\partial L}{\partial \alpha_i} = M \Big(\Psi\Big(\sum_{j=1}^{k} \alpha_i \Big) - \Psi(\alpha_i) \Big) + \sum_{d=1}^{M} \Big(\Psi(\gamma_{di}) - \Psi\Big(\sum_{j=1}^{k} \gamma_{dj} \Big)\Big)$$

This derivative depends on α_i, where $j \neq i$, and we therefore must use an iterative method to find the maximal α by linear-scaling Newton-Rhapson algorithm. In particular, the Hessian is in the form:

$$\frac{\partial^2 L}{\partial^2 \alpha_i} = \delta(i,j) M \Psi'(\alpha_i) - \Psi'\Big(\sum_{j=1}^{k} \alpha_i \Big). \tag{4.2}$$

5 Why LDA works?

Allowing topics sparsity and penalize too many topics of a document:

- When $\alpha < 1$, the majority of the probability mass is in the two side-edges of the distribution, generating mostly documents that have only a small number of topics.
- When $\alpha > 1$, most documents will have almost every topic.

Algorithm 1 Variational Expectation-Maximization for LDA

Input: k topics, M documents and N_d words.
Output: Model parameter: β, θ, z

```
1: Initialize: $\phi^0_{ni} := 1/k$ for all $i$ in $k$ and $n$ in $N_d$
2: Initialize: $\gamma_i := \alpha + N/k$ for all $i$ in $k$
3: Initialize: $\alpha_i := 50/k$
4: Initialize: $\beta_{ij} := 0$ for all $i$ in $k$ and $j$ in $V$
   //E-step: optimizing values of the variational parameters $\gamma_d, \phi_d$
5: loglikelihood := 0
6: for $d = 1$ to $M$ do
7:     repeat
8:         for $n = 1$ to $N$ do
9:             for $i = 1$ to $k$ do
10:                $\phi^{t+1}_{ni} := \beta_{iw_n} \exp(\Psi(\gamma^t_i))$
11:            end for
12:            normalize $\phi^{t+1}_n$ to sum to 1
13:        end for
14:        $\gamma^{t+1} := \alpha + \sum_{n=1}^{N} \phi^{t+1}_n$
15:    until convergence of $\gamma_d, \phi_d$
16:    loglikelihood: = loglikelihood + $L(\gamma, \phi; \alpha, \beta)$
17: end for
        //M-step: maximizing the lower bound with respect to parameters $\alpha, \beta$
18: for $n = 1$ to $M$ do
19:     for $i = 1$ to $k$ do
20:         for $j = 1$ to $V$ do
21:             $\beta_{ij} := \phi_{dni} w_{dnj}$
22:         end for
23:         normalize $\beta_i$ to sum to 1
24:     end for
25: end for
26: estimate $\alpha$ by (4.2)
27: if loglikelihood converged then
28:     return parameters
29: else
30:     go back to E-step
31: end if
```

LDA posterior put "topical" words together because of word co-occurences:

- Word probabilities are maximized by dividing the words among the topics. (More terms means more mass to be spread around.)
- In a mixture, this is enough to find clusters of co-occurring words.

The joint distribution allows our above statements to happen:

$$p(\theta, \boldsymbol{z} | \boldsymbol{w}, \alpha, \beta) \propto p(\theta, \boldsymbol{z} | \boldsymbol{w} | \alpha, \beta) = \underbrace{p(\boldsymbol{w} | \boldsymbol{z}, \beta)}_{1} \underbrace{p(\boldsymbol{z} | \theta)}_{2} \underbrace{p(\theta | \alpha)}_{3}$$

1. Encourage a sparse β in order to increase the probability of certain words w.
2. Encourage a sparse θ in order to increase the probability of certain topic z.
3. Using a small α will increase the probability of θ.

(1) wants to give form a sparse word clusters, (2) and (3) want to give a small number of topics to a document. The joint distribution encourages a small number of topics but the data itself needs a larger number of topics in order to assign to each document and also to form the diverse topics of all documents. So, as desired, we have a push and pull factor between the joint distribution and the actual data in determining the assignment of topics.

6 LDA's code implementation

The code is written in C language, Copyright 2004, David M. Blei (blei [at] cs [dot] cmu [dot] edu) and forked from GitHub.

The code will be explained by top down manner. The main program is directly straighforward, run EM or just inference.

Listing 4: **lda-estimate.c**

```c
int main(int argc, char* argv[])
{
   // (est / inf) alpha k settings data (random / seed/ model) (directory / out)

   corpus* corpus;

   long t1;
   (void) time(&t1);
   seedMT(t1);
   // seedMT(4357U);

   if (argc > 1)
   {
      if (strcmp(argv[1], "est")==0)
      {
         INITIAL_ALPHA = atof(argv[2]);
```

```
            NTOPICS = atoi(argv[3]);
            read_settings(argv[4]);
            corpus = read_data(argv[5]);
            make_directory(argv[7]);
            run_em(argv[6], argv[7], corpus);
        }
        if (strcmp(argv[1], "inf")==0)
        {
            read_settings(argv[2]);
            corpus = read_data(argv[4]);
            infer(argv[3], argv[5], corpus);
        }
    }
    else
    {
        printf("usage : lda est [initial alpha] [k] [settings] [data]
             [random/seeded/manual=filename/*] [directory]\n");
        printf("      lda inf [settings] [model] [data] [name]\n");
    }
    return(0);
}
```

EM algorithm is straighforward, here we only include important lines, for more details you can look at the entire lda-estimate.c . This step corresponds to line 26-29 in Algorithm 1:

Listing 5: **lda-estimate.c**

```
void run_em(char* start, char* directory, corpus* corpus)
{

    int d, n;
    lda_model *model = NULL;
    double **var_gamma, **phi;
    int i = 0;
    double likelihood, likelihood_old = 0, converged = 1;
    sprintf(filename, "%s/likelihood.dat", directory);
    FILE* likelihood_file = fopen(filename, "w");

    while (((converged < 0) || (converged > EM_CONVERGED) || (i <= 2)) && (i <=
        EM_MAX_ITER))
    {
        i++; printf("**** em iteration %d ****\n", i);
        likelihood = 0;
        zero_initialize_ss(ss, model);

        // e-step

        for (d = 0; d < corpus->num_docs; d++)
        {
            if ((d % 1000) == 0) printf("document %d\n",d);
            likelihood += doc_e_step(&(corpus->docs[d]),
                                      var_gamma[d],
                                      phi,
                                      model,
                                      ss);
        }

        // m-step
        lda_mle(model, ss, ESTIMATE_ALPHA);

        // check for convergence

        converged = (likelihood_old - likelihood) / (likelihood_old);
        if (converged < 0) VAR_MAX_ITER = VAR_MAX_ITER * 2;
        likelihood_old = likelihood;
```

This step corresponds to line 16 in Algorithm 1, after the convergence of γ_d, ϕ_d:

Listing 6: **lda-estimate.c**

```
double doc_e_step(document* doc, double* gamma, double** phi,
                  lda_model* model, lda_suffstats* ss)
{
    double likelihood;
    int n, k;

    // posterior inference

    likelihood = lda_inference(doc, model, gamma, phi);

    // update sufficient statistics
    // Find alpha_suffstats log$\theta_i$ to calculate \phi_{ni} \propto
        \beta_{iw_n}\text{exp}\{\text{E}_q[\text{log}\theta_i|\gamma]\}
    double gamma_sum = 0;
    for (k = 0; k < model->num_topics; k++)
    {
        gamma_sum += gamma[k];
        ss->alpha_suffstats += digamma(gamma[k]);
    }
    ss->alpha_suffstats -= model->num_topics * digamma(gamma_sum);

    for (n = 0; n < doc->length; n++)
    {
        for (k = 0; k < model->num_topics; k++)
        {
            ss->class_word[k][doc->words[n]] += doc->counts[n]*phi[n][k];
            ss->class_total[k] += doc->counts[n]*phi[n][k];
        }
    }

    ss->num_docs = ss->num_docs + 1;

    return(likelihood);
}
```

This step corresponds to line 6-15 in Algorithm 1:

Listing 7: **lda-inference.c**

```
double lda_inference(document* doc, lda_model* model, double* var_gamma,
     double** phi)
{
    double converged = 1;
    double phisum = 0, likelihood = 0;
    double likelihood_old = 0, oldphi[model->num_topics];
    int k, n, var_iter;
    double digamma_gam[model->num_topics];

    // compute posterior dirichlet
    // Initialize var_gamma_0, phi_0, digamma_gam_0
    for (k = 0; k < model->num_topics; k++)
    {
        var_gamma[k] = model->alpha + (doc->total/((double) model->num_topics));
             // \gamma_i := \alpha + N/k
        digamma_gam[k] = digamma(var_gamma[k]); // Init value for
             \phi_{ni}^{t+1}:=\beta_{iw_n} \text{exp}(\Psi(\gamma_i^t))
        for (n = 0; n < doc->length; n++)
            phi[n][k] = 1.0/model->num_topics; // \phi^0_{ni}:= 1/k
    }
    var_iter = 0;

    while ((converged > VAR_CONVERGED) &&
           ((var_iter < VAR_MAX_ITER) || (VAR_MAX_ITER == -1)))
    {
   var_iter++;
   for (n = 0; n < doc->length; n++)
   {
            phisum = 0;
            for (k = 0; k < model->num_topics; k++)
            {
                oldphi[k] = phi[n][k];
                phi[n][k] =
                    digamma_gam[k] +
                    model->log_prob_w[k][doc->words[n]];

                if (k > 0)
                    phisum = log_sum(phisum, phi[n][k]);
                else
                    phisum = phi[n][k]; // note, phi is in log space
            }

            for (k = 0; k < model->num_topics; k++)
            {
                phi[n][k] = exp(phi[n][k] - phisum);
                var_gamma[k] =
                    var_gamma[k] + doc->counts[n]*(phi[n][k] - oldphi[k]);
                // !!! a lot of extra digamma's here because of how we're
                     computing it
                // !!! but its more automatically updated too.
                digamma_gam[k] = digamma(var_gamma[k]);
            }
        }

        likelihood = compute_likelihood(doc, model, phi, var_gamma);
        assert(!isnan(likelihood));
        converged = (likelihood_old - likelihood) / likelihood_old;
        likelihood_old = likelihood;

        // printf("[LDA INF] %8.5f %1.3e\n", likelihood, converged);
    }
    return(likelihood);
}
```

This step calculate $L(\gamma, \phi; \alpha, \beta)$:

Listing 8: **lda-inference.c**

```
/*
 * compute likelihood bound
 *
 */

double
compute_likelihood(document* doc, lda_model* model, double** phi, double*
    var_gamma)
{
   double likelihood = 0, digsum = 0, var_gamma_sum = 0, dig[model->num_topics];
   int k, n;

   for (k = 0; k < model->num_topics; k++)
   {
  dig[k] = digamma(var_gamma[k]);
  var_gamma_sum += var_gamma[k];
   }
   digsum = digamma(var_gamma_sum);

   likelihood =
  lgamma(model->alpha * model -> num_topics)
  - model -> num_topics * lgamma(model->alpha)
  - (lgamma(var_gamma_sum));

   for (k = 0; k < model->num_topics; k++)
   {
  likelihood +=
      (model->alpha - 1)*(dig[k] - digsum) + lgamma(var_gamma[k])
      - (var_gamma[k] - 1)*(dig[k] - digsum);

  for (n = 0; n < doc->length; n++)
  {
          if (phi[n][k] > 0)
          {
              likelihood += doc->counts[n]*
                  (phi[n][k]*((dig[k] - digsum) - log(phi[n][k])
                             + model->log_prob_w[k][doc->words[n]]));
          }
      }
   }
   return(likelihood);
}
```

By understanding the above code, the reader can walk through the other files and the entire source code of the implementation without much difficulty.

Appendix A

A.1 Calculate $\mathbf{E}[\mathbf{log}\theta_i|\alpha]$

Since the LDA model assume $\theta \sim \mathrm{Dir}(\alpha)$, we have:

$$p(\theta|\alpha) = \frac{\Gamma\big(\sum_{i=1}^{k}\alpha_i\big)}{\prod_{i=1}^{k}\Gamma(\alpha_i)}\prod_{i=1}^{k}\theta_i^{\alpha_i-1}, \tag{6.1}$$

where the parameter α is a k-vector with components $\alpha_i > 0$, $\Gamma(x)$ is the Gamma function. Rewrite this by exponentiating the log:

$$p(\theta|\alpha) = \exp\Big\{\Big(\sum_{i=1}^{k}(\alpha_i - 1)\log\theta_i\Big) + \log\Gamma\Big(\sum_{i=1}^{k}\alpha_i\Big) - \sum_{i=1}^{k}\log\Gamma(\alpha_i)\Big\} \tag{6.2}$$

We can see that $p(\theta|\alpha)$ belongs to the exponential family as it can be written under this form:

$$p(x|\eta) = h(x)\exp\big\{\eta^T T(x) - A(\eta)\big\}$$

With $T(\theta_i) = \log(\theta_i), \eta_i = \alpha_i - 1, A(\eta_i) = \sum_{i=1}^{k}\log\Gamma(\alpha_i) - \log\Gamma\Big(\sum_{i=1}^{k}\alpha_i\Big)$
Using the properties of the cumulant generating function, in general we have $\mathrm{E}[T(x)] = \frac{d}{d\eta}A(\eta)$ (see proof). However, we're only interested in $T(\theta_i) = \log(\theta_i)$, therefore consider $\mathrm{E}[T(\theta_i)] = \frac{d}{d\eta_i}A(\eta_i)$:

$$\begin{aligned}\mathrm{E}[\log\theta_i|\alpha] &= \frac{d}{d\alpha_i}A(\alpha_i), \\ &= \frac{d}{d\alpha_i}\left(\sum_{i=1}^{k}\log\Gamma(\alpha_i) - \log\Gamma\Big(\sum_{i=1}^{k}\alpha_i\Big)\right),\end{aligned} \tag{6.3}$$

We obtain:

$$\mathrm{E}[\log\theta_i|\alpha] = \Psi(\alpha_i) - \Psi\Big(\sum_{j=1}^{k}\alpha_j\Big). \tag{6.4}$$

where $\Psi = \frac{d}{dx}\log(\Gamma(x))$, the digamma function.