Distributed Gibbs Sampling of Latent Dirichlet Allocation: The Gritty Details

Yi Wang
yiwang@tencent.com

THIS IS AN EARLY DRAFT. YOUR FEEDBACKS ARE HIGHLY APPRECIATED.

1 Introduction

I started to write this chapter since November 2007, right after my first MapReduce implementation of the AD-LDA algorithm[?]. I had worked so hard to understand LDA, but cannot find any document that were comprehensive, complete, and contain all necessary details. It is true that I can find some open source implementations of LDA, for example, LDA Gibbs sampler in Java[1] and GibbsLDA++[2], but code does not reveal math derivations. I read Griffiths' paper [?] and technical report [?], but realized most derivations are skipped. It was lucky to me that in the paper about Topic-over-time[?], an extension of LDA, the authors show part of the derivation. The derivation is helpful to understand LDA but is too short and not self-contained. A primer [?] by Gregor Heinrich contains more details than documents mentioned above. Indeed, most of my initial understanding on LDA comes from [?]. As [?] focuses more on the Bayesian background of latent variable modeling, I wrote this article focusing on more on the derivation of the Gibbs sampling algorithm for LDA. You may notice that the Gibbs sampling updating rule we derived in (38) differs slightly from what was presented in [?] and [?], but matches that in [?].

2 LDA and Its Learning Problem

In the literature, the training documents of LDA are often denoted by a long and segmented vector W:

$$W = \begin{bmatrix} \{w_1, \ldots, w_{N_1}\}, & \text{words in the 1st document} \\ \{w_{N_1+1}, \ldots, w_{N_1+N_2}\}, & \text{words in the 2nd document} \\ \cdots & \cdots \\ \{w_{1+\sum_{j=1}^{D-1} N_j}, \ldots, w_{\sum_{j=1}^{D} N_j}\} & \text{words in the } D\text{-th document} \end{bmatrix}, \tag{1}$$

where N_d denotes the number of words in the d-th document. [3]

In rest of this article, we consider hidden variables

$$Z = \begin{bmatrix} \{z_1, \ldots, z_{N_1}\}, & \text{topics of words in the 1st document} \\ \{z_{N_1+1}, \ldots, z_{N_1+N_2}\}, & \text{topics of words in the 2nd document} \\ \cdots & \cdots \\ \{z_{1+\sum_{j=1}^{D-1} N_j}, \ldots, z_{\sum_{j=1}^{D} N_j}\} & \text{topics of words in the } D\text{-th document} \end{bmatrix}, \tag{3}$$

where each $z_i \in Z$ corresponds to a word $w_i \in W$ in (1).

[1] http://arbylon.net/projects/LdaGibbsSampler.java
[2] http://gibbslda++.sourceforge.net
[3] Another representation of the training documents are two vectors: $\boldsymbol{d}$ and W:

$$\begin{bmatrix} \boldsymbol{d} \\ W \end{bmatrix} = \begin{bmatrix} d_1, & \ldots, & d_N \\ w_1, & \ldots, & w_N \end{bmatrix}, \tag{2}$$

where $N = \sum_{j=1}^{D} N_j$, d_i indices in $\mathcal{D}$, w_i indices in $\mathcal{W}$, and the tuple $[d_i, w_i]^T$ denotes an edge between the two vertices indexed by d_i and w_i respectively. Such representation is comprehensive and is used in many implementations of LDA, e.g, [?].

However, many papers (including this article) do not use this representation, because it misleads readers to consider two sets of random variables, W and $\boldsymbol{d}$. The fact is that $\boldsymbol{d}$ is the structure of W and is highly dependent with W. $\boldsymbol{d}$ is also the structure of the hidden variables Z, as shown by (3).

3 Dirichlet and Multinomial

To derive the Gibbs sampling algorithm with LDA, it is important to be familiar with the conjugacy between Dirichlet distribution and multinomial distribution.

The Dirichlet distribution is defined as:

$$\mathrm{Dir}(\boldsymbol{p};\boldsymbol{\alpha}) = \frac{1}{\mathrm{B}(\boldsymbol{\alpha})}\prod_{v=1}^{|\boldsymbol{\alpha}|} p_t^{\alpha_t - 1} \quad , \tag{4}$$

where the normalizing constant is the multinomial beta function, which can be expressed in terms of gamma function:

$$\mathrm{B}(\boldsymbol{\alpha}) = \frac{\prod_{i=1}^{|\boldsymbol{\alpha}|}\Gamma(\alpha_i)}{\Gamma(\sum_{i=1}^{|\boldsymbol{\alpha}|}\alpha_i)} \quad . \tag{5}$$

When Dirichlet distribution is used in LDA to model the prior, α_i's are positive integers. In this case, the gamma function degenerates to the factorial function:

$$\Gamma(n) = (n-1)! \tag{6}$$

The multinomial distribution is defined as:

$$\mathrm{Mult}(\boldsymbol{x};\boldsymbol{p}) = \frac{n!}{\prod_{i=1}^{K} x_i!}\prod_{i=1}^{K} p_i^{x_i} \quad , \tag{7}$$

where x_i denotes the number of times that value i appears in the samples drawn from the discrete distribution $\boldsymbol{p}$, $K = |\boldsymbol{x}| = |\boldsymbol{p}| = |\boldsymbol{\alpha}|$ and $n = \sum_{i=1}^{K} x_i$. The normalization constant comes from the product of combinatorial numbers.

We say the Dirichlet distribution is the conjugate distribution of the multinomial distribution, because if the prior of $\boldsymbol{p}$ is $\mathrm{Dir}(\boldsymbol{p};\boldsymbol{\alpha})$ and $\boldsymbol{x}$ is generated by $\mathrm{Mult}(\boldsymbol{x};\boldsymbol{p})$, then the posterior distribution of $\boldsymbol{p}$, $p(\boldsymbol{p}|\boldsymbol{x},\boldsymbol{\alpha})$, is a Dirichlet distribution:

$$\begin{aligned} p(\boldsymbol{p}|\boldsymbol{x},\boldsymbol{\alpha}) &= \mathrm{Dir}(\boldsymbol{p};\boldsymbol{x}+\boldsymbol{\alpha}) \\ &= \frac{1}{\mathrm{B}(\boldsymbol{x}+\boldsymbol{\alpha})}\prod_{v=1}^{|\boldsymbol{\alpha}|} p_t^{x_t+\alpha_t-1} \quad . \end{aligned} \tag{8}$$

Because (8) is a probability distribution function, integrating it over $\boldsymbol{p}$ should result in 1:

$$\begin{aligned} 1 &= \int \frac{1}{\mathrm{B}(\boldsymbol{x}+\boldsymbol{\alpha})}\prod_{v=1}^{|\boldsymbol{\alpha}|} p_t^{x_t+\alpha_t-1}\mathrm{d}\boldsymbol{p} \\ &= \frac{1}{\mathrm{B}(\boldsymbol{x}+\boldsymbol{\alpha})}\int \prod_{v=1}^{|\boldsymbol{\alpha}|} p_t^{x_t+\alpha_t-1}\mathrm{d}\boldsymbol{p} \quad . \end{aligned} \tag{9}$$

This implies

$$\int \prod_{v=1}^{|\boldsymbol{\alpha}|} p_t^{x_t+\alpha_t-1}\mathrm{d}\boldsymbol{p} = \mathrm{B}(\boldsymbol{x}+\boldsymbol{\alpha}) \quad . \tag{10}$$

This property will be used in the following derivations.

4 Learning LDA by Gibbs Sampling

There have been three strategies to learn LDA: EM with variational inference [**?**], EM with expectation propagation [**?**], and Gibbs sampling [**?**]. In this article, we focus on the Gibbs sampling method, whose performance is comparable with the other two but is tolerant better to local optima.

Gibbs sampling is one of the class of samplings methods known as Markov Chain Monte Carlo. We use it to sample from the posterior distribution, $p(Z|W)$, given the training data W represented in the form of (1). As will be shown by the following text, given the sample of Z we can infer model parameters $\boldsymbol{\Phi}$ and $\boldsymbol{\Theta}$. This forms a learning algorithm with its general framework shown in Fig. 1.

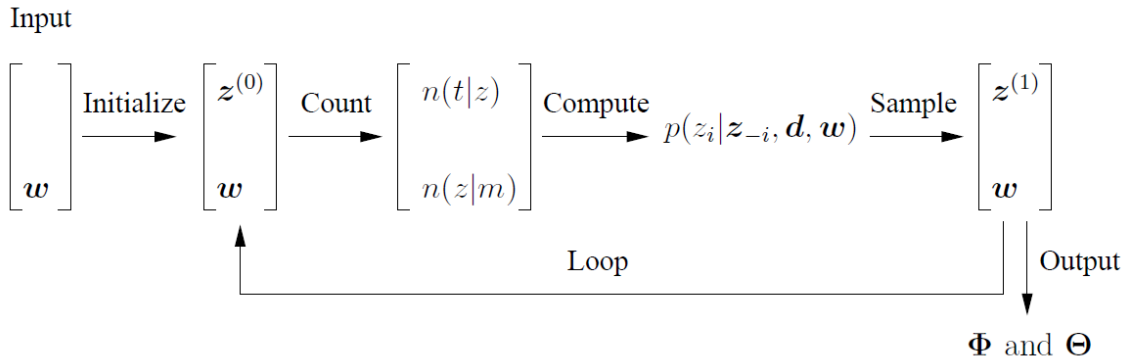

Figure 1: The procedure of learning LDA by Gibbs sampling.

N	the number of words in the corpus
$1 \leq i, j \leq N$	index of words in the corpus
$W = \{w_i\}$	the corpus, and w_i denotes a word
$Z = \{z_i\}$	latent topics assigned to words in W
$W^{\neg i} = W \setminus w_i$	the corpus excluding w_i
$Z^{\neg i} = Z \setminus z_i$	latent topics excluding z_i
K	the number of topics specified as a parameter
V	the number of unique words in the vocabulary
$\boldsymbol{\alpha}$	the parameters of topic Dirichlet prior
$\boldsymbol{\beta}$	the parameters of word Dirichlet prior
$\Omega_{d,k}$	count of words in d assigned topic k; Ω_d denotes the d-th row of matrix Ω.
$\Psi_{k,v}$	count of word v in corpus assigned k Ψ_k denotes the k-th row of matrxi Ψ.
$\Omega^{\neg i}_{d,k}$	like $\Omega_{d,k}$ but excludes w_i and z_i
$\Psi^{\neg i}_{k,v}$	like $\Psi_{k,v}$ but excludes w_i and z_i
$\boldsymbol{\Theta} = \{\theta_{d,k}\}$	$\theta_{d,k} = P(z = k\|d)$, $\boldsymbol{\theta}_d = P(z\|d)$.
$\boldsymbol{\Phi} = \{\phi_{k,v}\}$	$\phi_{k,v} = P(w = v\|z = k)$, $\boldsymbol{\phi}_k = P(w\|z = k)$.

Table 1: Symbols used in the derivation of LDA Gibbs sampling rule.

In order to sample from $p(Z|W)$ using the Gibbs sampling method, we need the full conditional posterior distribution $p(z_i|Z^{\neg i}, W)$, where $Z^{\neg i}$ denotes all z_j's with $j \neq i$. In particular, Gibbs sampling does not require knowing the exact form of $p(z_i|Z^{\neg i}, W)$; instead, it is enough to have a function $f(\cdot)$, where

$$f(z_i|Z^{\neg i}, W) \propto p(z_i|Z^{\neg i}, W) \tag{11}$$

The following mathematical derivation is for such a function $f(\cdot)$.

The Joint Distribution of LDA We start from deriving the joint distribution [4],

$$p(Z, W|\boldsymbol{\alpha}, \boldsymbol{\beta}) = p(W|Z, \boldsymbol{\beta})p(Z|\boldsymbol{\alpha}) \ , \tag{12}$$

which is the basis of the derivation of the Gibbs updating rule and the parameter estimation rule. As $p(W|Z, \boldsymbol{\beta})$ and $p(Z|\boldsymbol{\alpha})$ depend on $\boldsymbol{\Phi}$ and $\boldsymbol{\Theta}$ respectively, we derive them separately.

According to the definition of LDA, we have

$$p(W|Z, \boldsymbol{\beta}) = \int p(W|Z, \boldsymbol{\Phi})p(\boldsymbol{\Phi}|\boldsymbol{\beta})\mathrm{d}\boldsymbol{\Phi} \ , \tag{13}$$

where $p(\boldsymbol{\Phi}|\boldsymbol{\beta})$ has Dirichlet distribution:

$$p(\boldsymbol{\Phi}|\boldsymbol{\beta}) = \prod_{k=1}^{K} p(\boldsymbol{\phi}_k|\boldsymbol{\beta}) = \prod_{k=1}^{K} \frac{1}{\mathrm{B}(\boldsymbol{\beta})} \prod_{v=1}^{V} \phi_{k,v}^{\beta_v - 1} \ , \tag{14}$$

[4] Notations used in this article are identical with those used in [?].

and $p(W|Z, \mathbf{\Phi})$ has multinomial distribution:

$$p(W|Z, \mathbf{\Phi}) = \prod_{i=1}^{N} \phi_{z_i, w_i} = \prod_{k=1}^{K} \prod_{v=1}^{V} \phi_{k,v}^{\Psi_{k,v}} \quad , \tag{15}$$

where Ψ is a $K \times V$ count matrix and $\Psi_{k,v}$ is the number of times that topic k is assigned to word v. With W and Z defined by (1) and (3) respectively, we can represent $\Psi_{k,v}$ mathematically by

$$\Psi_{k,v} = \sum_{i=1}^{N} \mathrm{I}\{w_i = v \wedge z_i = k\} \quad , \tag{16}$$

where N is the corpus size in number of words. In later sections, we use Ψ_k to denote the k-th row of the matrix Ψ.

Given (15) and (14), (13) becomes

$$p(W|Z, \boldsymbol{\beta}) = \int \prod_{k=1}^{K} \frac{1}{\mathrm{B}(\boldsymbol{\beta})} \prod_{v=1}^{V} \phi_{k,v}^{\Psi_{k,v}+\beta_v-1} \mathrm{d}\boldsymbol{\phi}_k \quad . \tag{17}$$

Using the property of integration of a product, which we learned in our colleague time,

$$\int \prod_{k=1}^{K} f_k(\phi_k) \, \mathrm{d}\phi_1 \ldots \mathrm{d}\phi_K = \prod_{k=1}^{K} \int f_k(\phi_k) \, \mathrm{d}\phi_k \quad , \tag{18}$$

we have

$$\begin{aligned} p(W|Z, \boldsymbol{\beta}) &= \prod_{k=1}^{K} \left(\int \frac{1}{\mathrm{B}(\boldsymbol{\beta})} \prod_{v=1}^{V} \phi_{k,v}^{\Psi_{k,v}+\beta_v-1} \mathrm{d}\boldsymbol{\phi}_k \right) \\ &= \prod_{k=1}^{K} \left(\frac{1}{\mathrm{B}(\boldsymbol{\beta})} \int \prod_{v=1}^{V} \phi_{k,v}^{\Psi_{k,v}+\beta_v-1} \mathrm{d}\boldsymbol{\phi}_k \right) \quad . \end{aligned} \tag{19}$$

Using property (10), we can compute the integratation over $\boldsymbol{\phi}_k$ in a close form, so

$$p(W|Z, \boldsymbol{\beta}) = \prod_{k=1}^{K} \frac{\mathrm{B}(\Psi_k + \boldsymbol{\beta})}{\mathrm{B}(\boldsymbol{\beta})} \quad , \tag{20}$$

where Ψ_k denotes the k-th row of matrix Ψ.

Now we derive $p(Z|\boldsymbol{\alpha})$ analogous to $p(W|Z, \boldsymbol{\beta})$. Similar with (14), we have

$$p(\mathbf{\Theta}|\boldsymbol{\alpha}) = \prod_{d=1}^{D} p(\boldsymbol{\theta}_m|\boldsymbol{\alpha}) = \prod_{d=1}^{D} \frac{1}{\mathrm{B}(\boldsymbol{\alpha})} \prod_{k=1}^{K} \theta_{d,k}^{\alpha_k-1} \quad ; \tag{21}$$

similar with (15), we have

$$p(Z|\mathbf{\Theta}) = \prod_{i=1}^{N} \theta_{d_i, z_i} = \prod_{d=1}^{D} \prod_{k=1}^{K} \theta_{d,k}^{\Omega_{d,k}} \quad ; \tag{22}$$

and similar with (20) we have

$$\begin{aligned} p(Z|\boldsymbol{\alpha}) &= \int p(Z|\mathbf{\Theta}) p(\mathbf{\Theta}|\boldsymbol{\alpha}) \mathrm{d}\mathbf{\Theta} \\ &= \prod_{d=1}^{D} \left(\int \frac{1}{\mathrm{B}(\boldsymbol{\alpha})} \prod_{k=1}^{K} \theta_{d,k}^{\Omega_{d,k}+\alpha_k-1} \mathrm{d}\boldsymbol{\theta}_d \right) \\ &= \prod_{d=1}^{D} \frac{\mathrm{B}(\Omega_d + \boldsymbol{\alpha})}{\mathrm{B}(\boldsymbol{\alpha})} \quad , \end{aligned} \tag{23}$$

where Ω is a count matrix and $\Omega_{d,k}$ is the number of times that topic k is assigned to words in document d; Ω_d denotes the d-th row of Ω. With input data defined by (2), $\Omega_{d,k}$ can be represented mathematically

$$\Omega_{d,k} = \sum_{i=1}^{N} \mathrm{I}\{d_i = d \wedge z_i = k\} \quad , \tag{24}$$

where N is the corpus size by words. In later sections, we use Ω_d to denote the d-th row of the matrix Ω.

Given (20) and (23), the joint distribution (12) becomes:

$$\begin{aligned} p(Z, W|\boldsymbol{\alpha}, \boldsymbol{\beta}) &= p(W|Z, \boldsymbol{\beta})p(Z|\boldsymbol{\alpha}) \\ &= \prod_{k=1}^{K} \frac{\mathrm{B}(\Psi_k + \boldsymbol{\beta})}{\mathrm{B}(\boldsymbol{\beta})} \cdot \prod_{d=1}^{D} \frac{\mathrm{B}(\Omega_d + \boldsymbol{\alpha})}{\mathrm{B}(\boldsymbol{\alpha})} \end{aligned} \tag{25}$$

The Gibbs Updating Rule With (25), we can derive the Gibbs updating rule for LDA:

$$p(z_i = k|Z^{\neg i}, W, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{p(z_i = k, Z^{\neg i}, W|\boldsymbol{\alpha}, \boldsymbol{\beta})}{p(Z^{\neg i}, W|\boldsymbol{\alpha}, \boldsymbol{\beta})} \ . \tag{26}$$

Because z_i depends only on w_i, (26) becomes

$$p(z_i|Z^{\neg i}, W, \boldsymbol{\alpha}, \boldsymbol{\beta}) \propto \frac{p(Z, W|\boldsymbol{\alpha}, \boldsymbol{\beta})}{p(Z^{\neg i}, W^{\neg i}|\boldsymbol{\alpha}, \boldsymbol{\beta})} \qquad \text{where } z_i = k \ . \tag{27}$$

The numerator in (27) is (25); whereas the denominator has a similar form:

$$\begin{aligned} p(Z^{\neg i}, W^{\neg i}|\boldsymbol{\alpha}, \boldsymbol{\beta}) &= p(W^{\neg i}|Z^{\neg i}, \boldsymbol{\beta})p(Z^{\neg i}|\boldsymbol{\alpha}) \\ &= \prod_{k=1}^{K} \frac{\mathrm{B}(\Psi_k^{\neg i} + \boldsymbol{\beta})}{\mathrm{B}(\boldsymbol{\beta})} \cdot \prod_{d=1}^{D} \frac{\mathrm{B}(\Omega_d^{\neg i} + \boldsymbol{\alpha})}{\mathrm{B}(\boldsymbol{\alpha})} \ , \end{aligned} \tag{28}$$

where $\Psi_k^{\neg i}$ denotes the k-th row of the count $K \times V$ count matrix $\Psi^{\neg i}$, and $\Psi_{k,v}^{\neg i}$ is the number of times that topic k is assigned to word w, but with the i-th word and its topic assignment excluded. Similarly, $\Omega_d^{\neg i}$ is the d-th row of count matrix $\Omega^{\neg i}$, and $\Omega_{d,k}^{\neg i}$ is the number of words in document d that are assigned topic k, but with the i-th word and its topic assignment excluded. In more details,

$$\begin{aligned} \Psi_{k,v}^{\neg i} &= \sum_{\substack{1 \le j \le N \\ j \neq i}} \mathrm{I}\{w_j = v \wedge z_j = k\} \ , \\ \Omega_{d,k}^{\neg i} &= \sum_{\substack{1 \le j \le N \\ j \neq i}} \mathrm{I}\{d_j = d \wedge z_j = k\} \ , \end{aligned} \tag{29}$$

where d_j denotes the document to which the i-th word in the corpus belongs. Some properties can be derived from the definitions directly:

$$\begin{aligned} \Psi_{k,v} &= \begin{cases} \Psi_{k,v}^{\neg i} + 1 & \text{if } v = w_i \text{ and } k = z_i; \\ \Psi_{k,v}^{\neg i} & \text{all other cases.} \end{cases} \\ \Omega_{d,k} &= \begin{cases} \Omega_{d,k}^{\neg i} + 1 & \text{if } d = d_i \text{ and } k = z_i; \\ \Omega_{d,k}^{\neg i} & \text{all other cases.} \end{cases} \end{aligned} \tag{30}$$

From (30) we have

$$\begin{aligned} \sum_{v=1}^{V} \Psi_{z_i,v} &= 1 + \sum_{v=1}^{V} \Psi_{z_i,v}^{\neg i} \\ \sum_{k=1}^{K} \Omega_{d_i,z} &= 1 + \sum_{k=1}^{K} \Omega_{d_i,z}^{\neg i} \end{aligned} \tag{31}$$

Using (30) and (31), we can simplify (28):

$$p(z_i = k|Z^{\neg i}, W, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{\mathrm{B}(\Psi_k + \boldsymbol{\beta})}{\mathrm{B}(\Psi_k^{\neg i} + \boldsymbol{\beta})} \cdot \frac{\mathrm{B}(\Omega_d + \boldsymbol{\alpha})}{\mathrm{B}(\Omega_d^{\neg i} + \boldsymbol{\alpha})} \ . \tag{32}$$

where d denotes the document that contains w_i.

Further simplification can be achieved by substituting

$$\mathrm{B}(\boldsymbol{x}) = \frac{\prod_{k=1}^{\dim \boldsymbol{x}} \Gamma(x_k)}{\Gamma(\sum_{k=1}^{\dim \boldsymbol{x}} x_k)} \ , \tag{33}$$

and we have

$$p(z_i = k|Z^{\neg i}, w_i = v, W^{\neg i}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{\prod_{v=1}^{V} \Gamma(\Psi_{k,v}+\beta_v)}{\Gamma(\sum_{v=1}^{V} \Psi_{k,v}+\beta_v)} \cdot \frac{\prod_{k=1}^{K} \Gamma(\Omega_{d,k}+\alpha_k)}{\Gamma(\sum_{k=1}^{K} \Omega_{d,k}+\alpha_k)} \cdot \frac{\prod_{v=1}^{V} \Gamma(\Psi^{\neg i}_{k,v}+\beta_v)}{\Gamma(\sum_{v=1}^{V} \Psi^{\neg i}_{k,v}+\beta_v)} \cdot \frac{\prod_{k=1}^{K} \Gamma(\Omega^{\neg i}_{d,k}+\alpha_k)}{\Gamma(\sum_{k=1}^{K} \Omega^{\neg i}_{d,k}+\alpha_k)} \ . \tag{34}$$

Using (30), we can remove most factors in the products of (34) and get

$$p(z_i = k|Z^{\neg i}, w_i = v, W^{\neg i}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{\Gamma(\Psi_{k,v}+\beta_v)}{\Gamma(\sum_{v=1}^{V} \Psi_{k,v}+\beta_v)} \cdot \frac{\Gamma(\Omega_{d,k}+\alpha_k)}{\Gamma(\sum_{k=1}^{K} \Omega_{d,k}+\alpha_k)} \cdot \frac{\Gamma(\Psi^{\neg i}_{k,v}+\beta_v)}{\Gamma(\sum_{v=1}^{V} \Psi^{\neg i}_{k,v}+\beta_v)} \cdot \frac{\Gamma(\Omega^{\neg i}_{d,k}+\alpha_k)}{\Gamma(\sum_{k=1}^{K} \Omega^{\neg i}_{d,k}+\alpha_k)} \ . \tag{35}$$

Here it is important to know that

$$\Gamma(y) = (y-1)! \qquad \text{if } y \text{ is a positive integer} \ , \tag{36}$$

so we expand (35) and using (30) to remove most factors in the factorials. This leads us to the Gibbs updating rule for LDA:

$$p(z_i = k|Z^{\neg i}, w = v, W^{\neg i}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{\Psi_{k,v} + \beta_{w_i} - 1}{\left[\sum_{v=1}^{V} \Psi_{k,v} + \beta_t\right] - 1} \cdot \frac{\Omega_{d,k} + \alpha_k - 1}{\left[\sum_{k=1}^{K} \Omega_{d,k} + \alpha_z\right] - 1} \ . \tag{37}$$

Note that the denominator of the second factor at the right hand side of (38) does not depend on z_i, the parameter of the function $p(z_i|Z^{\neg i}, W, \boldsymbol{\alpha}, \boldsymbol{\beta})$. Because the Gibbs sampling method requires only a function $f(z_i) \propto p(z_i)$, c.f. (11), we can use the following updating rule in practice:

$$p(z_i = k|Z^{\neg i}, w = v, W^{\neg i}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{\Psi_{k,v} + \beta_{w_i} - 1}{\left[\sum_{v=1}^{V} \Psi_{k,v} + \beta_t\right] - 1} \cdot \left[\Omega_{d,k} + \alpha_k - 1\right] \ . \tag{38}$$

Parameter Estimation By the definition of $\phi_{k,v}$ and $\theta_{d,k}$, we have

$$\begin{aligned} \phi_{k,v} &= \frac{\Psi_{k,v} + \beta_t}{\left(\sum_{v'=1}^{V} \Psi_{k,v'} + \beta_{v'}\right)} \ , \\ \theta_{m,k} &= \frac{\Omega_{d,k} + \alpha_k}{\left(\sum_{k=1}^{K} \Omega_{d,k} + \alpha_z\right)} \ . \end{aligned} \tag{39}$$

The Algorithm With (38), we can realize the learning procedure shown in Fig. 1 using Algorithm. 1.

In this algorithm, we use a 2D array `NWZ[w,z]` to maintain Ψ and `NZM[z,m]` for Ω. Also, to accelerate the computation, we use a 1D array `NZ[z]` to maintain $n(z) = \sum_{v=1}^{|\mathcal{W}|} n(t; z)$. Note that we do not need an array `NM[d]` for $n(z) = \sum_{k=1}^{|\mathcal{Z}|} n(z, d)$. Although it appears in (37), but it is not necessary in (38).

The input of this algorithm is not in the form of (1); instead, it is in a more convenient form: with each document represented by a set of words it includes, the input is given as a set of documents. Denote the n-th word in the m-th document by $w_{m,n}$, the corresponding topic is denoted by $z_{m,n}$ and corresponding document is m. With these correspondence known, the above derivations can be easily implemented in the algorithm.

zero all count variables `NWZ`, `NZM`, `NZ` ;
foreach *document* $m \in [1, D]$ **do**
  foreach *word* $n \in [1, N_m]$ *in document* m **do**
    sample topic index $z_{m,n} \sim \text{Mult}(1/K)$ for word $w_{m,n}$;
    increment document-topic count: `NZM[`$z_{m,n}, m$`]++` ;
    increment topic-term count: `NWZ[`$w_{m,n}, z_{m,n}$`]++` ;
    increment topic-term sum: `NZ[`$z_{m,n}$`]++` ;
  end
end
while *not finished* **do**
  foreach *document* $m \in [1, D]$ **do**
    foreach *word* $n \in [1, N_m]$ *in document* m **do**
      `NWZ[`$w_{m,n}, z_{m,n}$`]--`, `NZ[`$z_{m,n}$`]--`, `NZM[`$z_{m,n}, m$`]--` ;
      sample topic index $\tilde{z}_{m,n}$ according to (40) ;
      `NWZ[`$w_{m,n}, \tilde{z}_{m,n}$`]++`, `NZ[`$\tilde{z}_{m,n}$`]++`, `NZM[`$\tilde{z}_{m,n}, m$`]++` ;
    end
  end
  if *converged and* L *sampling iterations since last read out* **then**
    read out parameter set $\boldsymbol{\Theta}$ and $\boldsymbol{\Phi}$ according to (39) ;
  end
end

Algorithm 1: The Gibbs sampling algorithm that learns LDA.

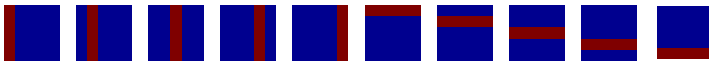

Figure 2: The ground-truth $\boldsymbol{\Phi}$ used to synthesize our testing data.

The sampling operation in this algorithm is not identical with the full conditional posterior distribution (38); instead, it does not contain those "-1" terms:

$$p(z_i) = \frac{\Psi_{z_i,w_i} + \beta_{w_i}}{\left[\sum_{v=1}^{V} \Psi_{k,v} + \beta_t\right]} \cdot \left[\Omega_{d_i,z_i} + \alpha_{z_i}\right] \ . \tag{40}$$

This makes it elegant to maintain the consistency of sampling new topics and updating the counts (`NWZ`, `NZ` and `NZM`) using three lines of code: decreasing the counts, sampling and increasing the counts. [5]

5 Experiments

The synthetic image data mentioned in [**?**] is useful to test the correctness of any implementation of LDA learning algorithm. To synthesize this data set, we fix $\boldsymbol{\Phi} = \{\boldsymbol{\phi}_k\}_{k=1}^{K=10}$ as visualized in Fig. 2 (also, Fig. 1 in [**?**]), set $\boldsymbol{\alpha} = [1]$, the number of words/pixels in each document/image $d = 100$. Because every image has 5×5 pixels, the vocabulary size is 25.[6]

Fig. 3 shows the result of running Gibbs sampling to learn an LDA from this testing data set, where the left pane shows the convergence of the likelihood, and the right pane shows the updating process of $\boldsymbol{\Phi}$ along the iterations of the algorithm. The 20 rows of images in the right pane are visualizations of $\boldsymbol{\Phi}$ estimated at the iterations of 1, 2, 3, 5, 7, , 10, 15, 20, 25, 50, 75, 100, 125, 150, 175, , 200, 250, 300, 400, 499. From this figure we can see that since iteration 100, the estimated $\boldsymbol{\Phi}$ is almost identical to Fig. 1(a) of [**?**].

[5] This trick illustrates the physical meaning of those "-1" terms in (38). I learn this trick from Heinrich's code at `http://arbylon.net/projects/LdaGibbsSampler.java`.

[6] All these settings are identical with those described in [**?**].

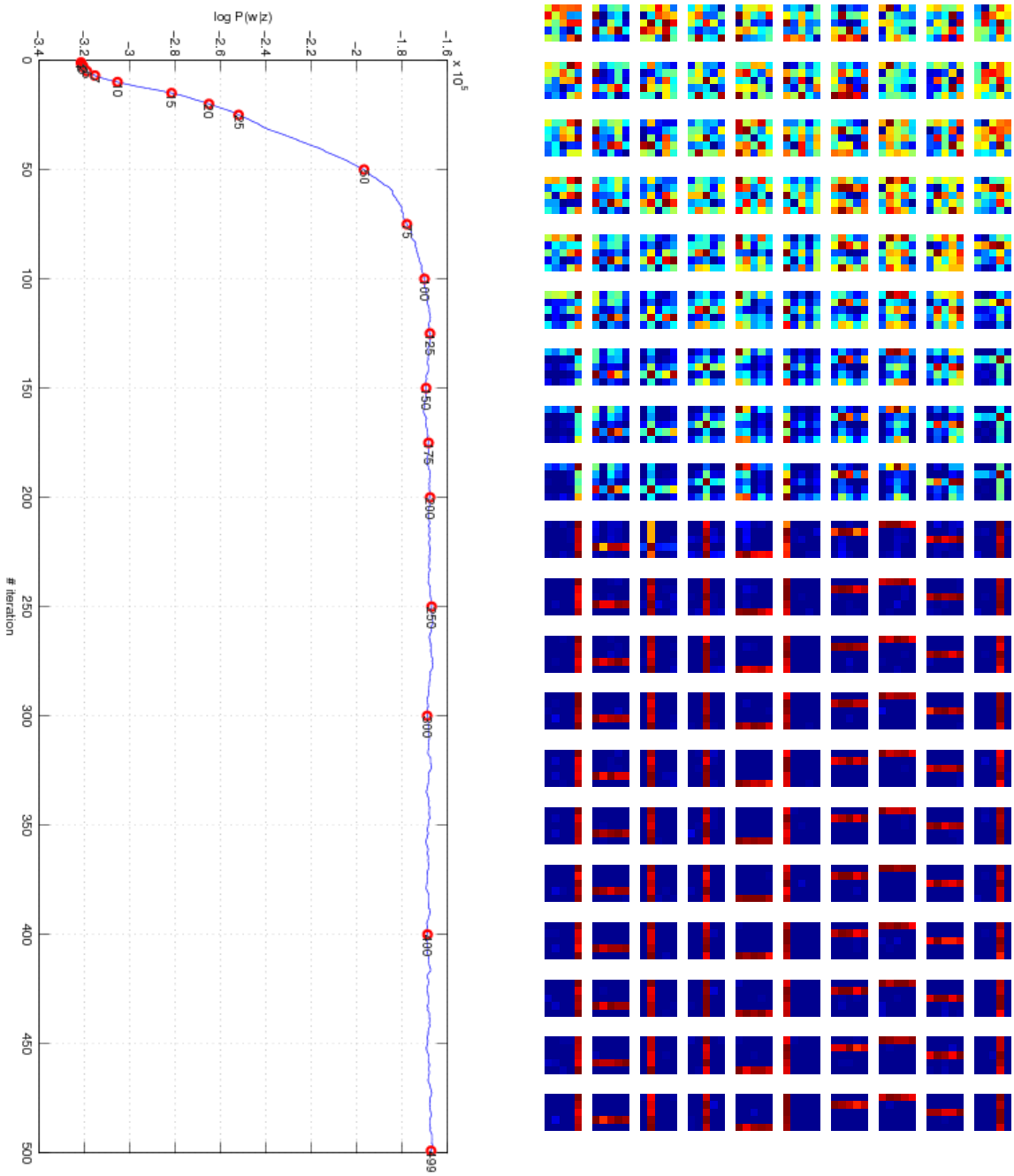

Figure 3: The result of running the Gibbs sampling algorithm to learn an LDA from synthetic data. Each row of 10 images in the right pane visualizes the $\mathbf{\Phi} = \{\boldsymbol{\phi}_k\}_{k=1}^{K=10}$ estimated at those iterations indicated by red circles in the left pane.

6 Acknowledgement

Thanks go to Gregor Heinrich and Xuerui Wang for their gentle explanation of some math details.