Gibbs sampling for LDA: understanding implementation

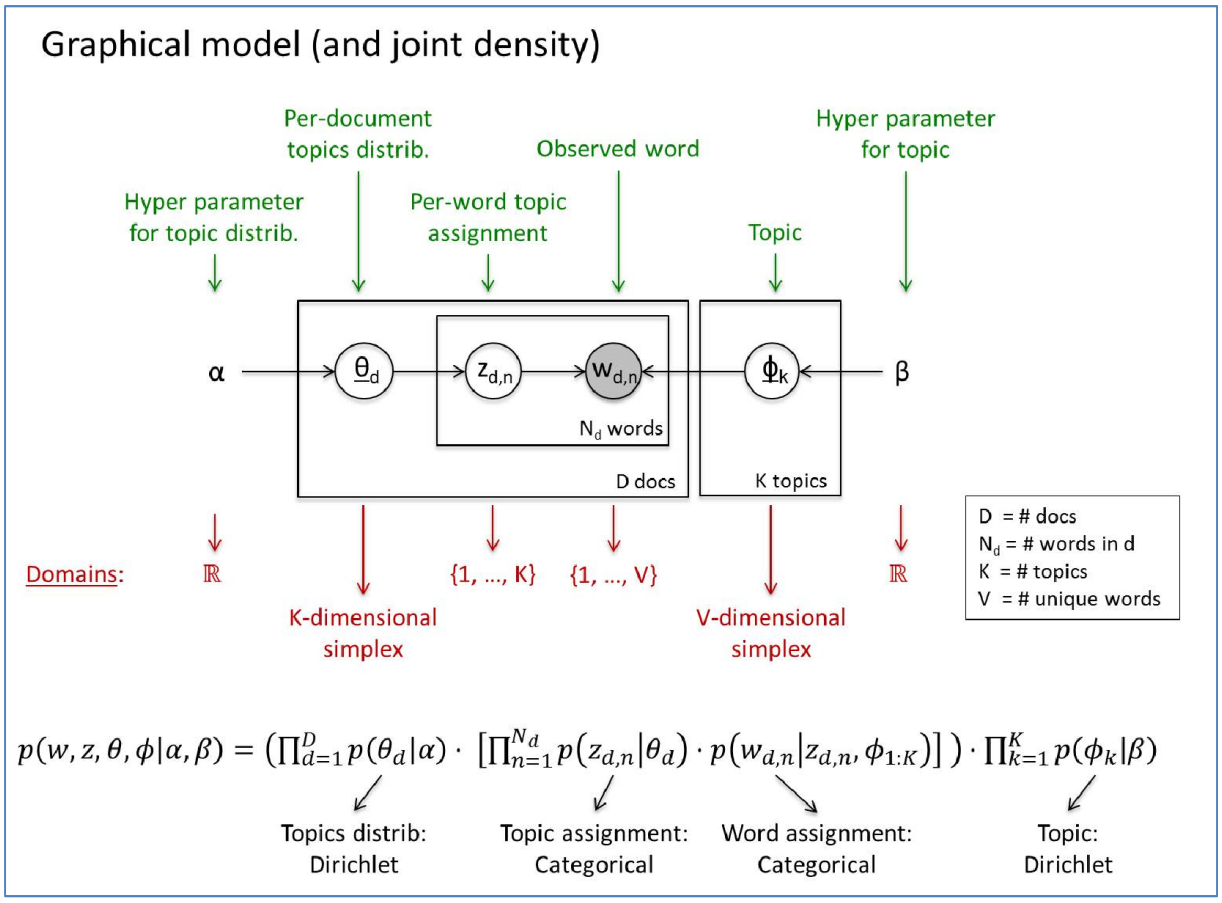
The goal of this document is to explain how LDA toolbox works. Particularly, it aims to link the LDA model and Gibbs sampling to the actual implementation, by describing precisely the data structures used.

We derive the Gibbs sampling algorithm for LDA and give the important equations along this derivation. It is not needed to understand these equations for a simple use of LDA, however they can be useful references when wanting to adapt the algorithm (e.g. for robust estimation or for supervised LDA).

For an introduction to LDA, a good reference is 'Probabilistic Topic Models' by Steyvers and Griffiths (2007).

LDA model

Here is a recall of how the LDA model usually introduced, in order to make link with following derivations. The model can be expressed by the graphical model or by the joint density below.



Inference

As we know the words, what we want to obtain is the distribution $p(z,\theta,\phi|w,\alpha,\beta)$, or at least its mode(s). However, this cannot be obtained exactly (intractable, i.e. computationally too expensive). Therefore, it has to be approximated, in the following using Gibbs sampling.

Practical notation for computations

We first need to define some notation reflecting the structures that will be used in practice.

Let $N = \sum N_d$ be the total number of tokens (= words instances) in all documents. The corpus can be described token by token using two vectors of length N: $\underline{w} \in \{1, \dots, V\}^N$ and $\underline{d} \in \{1, \dots, D\}^N$. The first vector describes the **word associated to each token**, and the second vector describes the **document in which each token appear**.

For example, suppose we have D=2 documents: doc1 = 'hello hello world', doc2 = 'brave new world'. We have a vocabulary of size V=4 and we can build a mapping (hello:1, world:2, brave:3, new:4). Then we would have $\underline{w} = [1\ 1\ 2\ 3\ 4\ 2]^T$ and $\underline{d} = [1\ 1\ 1\ 2\ 2\ 2]^T$.

For Gibbs sampling, we will also need a third vector of length N: $\underline{z} \in \{1, \dots, K\}^N$. This third vector contains the **topic assigned to each token**.

For example if all tokens of doc1 are assigned topic 2, and if in doc2 'brave' and 'world' are assigned topic 1 while new is assigned to topic 2, then we would have $\underline{z} = [2\ 2\ 2\ 1\ 2\ 1]^T$.

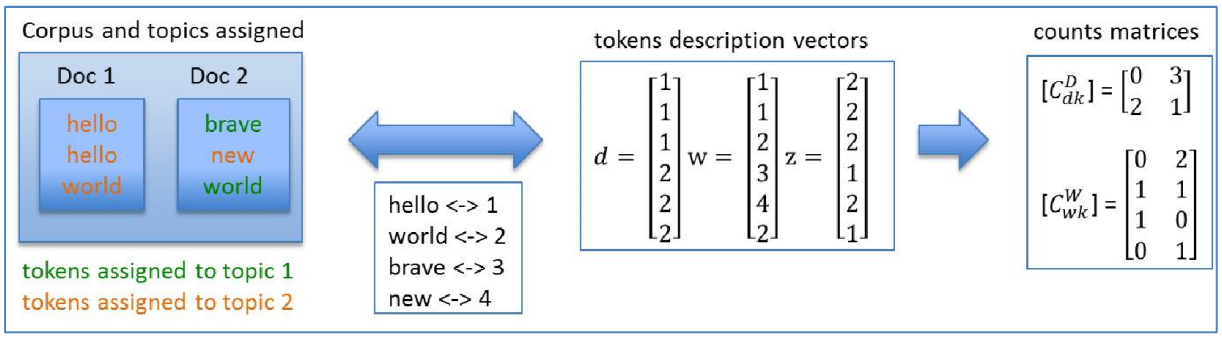
Using this new notation, we can rewrite the joint density:

$$p(w, z, \theta, \phi | \alpha, \beta) = \left(\prod_{d=1}^{D} p(\theta_d | \alpha) \cdot \left[\prod_{n=1}^{N_d} p(z_{d,n} | \theta_d) \cdot p(w_{d,n} | z_{d,n}, \phi_{1:K}) \right] \right) \cdot \prod_{k=1}^{K} p(\phi_k | \beta)$$

in the equivalent form:

$$p(w, z, \theta, \phi | \alpha, \beta) = \left[\prod_{i=1}^{N} p(z_i | \theta_{d_i}) \cdot p(w_i | z_i, \phi_{1:K}) \right] \prod_{d=1}^{D} p(\theta_d | \alpha) \cdot \prod_{k=1}^{K} p(\phi_k | \beta)$$

We cannot anymore see the direct link between the joint density and the graphical model, however the new notation is closer to what will be used for computations. The picture below summarizes this notation (count matrices are introduced in the next section).



Integrating out θ and ϕ

It is possible to obtain an exact expression for $p(z, w | \alpha, \beta)$, by using conjugate properties of dirichlet:

$$p(z, w | \alpha, \beta) = p(z | \alpha) \times p(w | z, \beta)$$

$$= \left(\frac{\Gamma(K\alpha)}{\Gamma(\alpha)^K} \right)^D \prod_{d=1}^{D} \frac{\prod_{k=1}^{K} \Gamma(\alpha + C_{dk}^D)}{\Gamma(K\alpha + \sum_{k=1}^{K} C_{dk}^D)} \times \left(\frac{\Gamma(V\beta)}{\Gamma(\beta)^V} \right)^K \prod_{k=1}^{K} \frac{\prod_{w=1}^{V} \Gamma(\beta + C_{wk}^W)}{\Gamma(V\beta + \sum_{w=1}^{V} C_{wk}^W)}$$

where Γ is the gamma function, and where we define the **counts matrices**:

- $C_{dk}^{D} = \#\{j \in \{1, \dots, N\}: d_j = d \; and \; z_j = k)$ is the **number of times topic k is assigned to document d** (for a given topic assignment $\underline{z}$)
- $C_{wk}^{W} = \#\{j \in \{1, \dots, N\}: \; w_j = w \; and \; z_j = k)$ is the **number of times topic k is assigned to word w** (for a given topic assignment $\underline{z}$)

$[C_{dk}^{D}]$ form a DxK matrix and $[C_{wk}^{W}]$ forms a VxK matrix, and these matrices are important in practice.

For the previous example, we would have $[C_{dk}^{D}] = \begin{bmatrix} 0 & 3 \\ 2 & 1 \end{bmatrix}$ and $[C_{wk}^{W}] = \begin{bmatrix} 0 & 2 \\ 1 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$.

$[C_{dk}^{D}]$ can be read '*In doc 1, no token was assigned to topic 1 and 3 tokens were assigned to topic 2; In doc 2, 2 tokens were assigned to topic 1 and 1 token was assigned to topic 2*'.

$[C_{wk}^{W}]$ can be read *'Word 1 was assigned 0 times to topic 1 and 2 times to topic 2, word 2 was assigned 1 time to topic 1 and 1 time to topic 2, …'*.

We cannot interpret as easily the new marginalized joint density $p(z, w|\alpha, \beta)$ (at least I can't). However this will allow to approximate the distribution $p(z|w, \alpha, \beta)$ rather than $p(z, \theta, \phi|w, \alpha, \beta)$.

Gibbs sampling of $p(z|w,\alpha,\beta)$

z is a high dimensional random vector (N dimensional), and as before exact inference is infeasible. Therefore we use the Gibbs sampling algorithm, which allows **obtaining samples z^s from the distribution $p(z|w,\alpha,\beta)$**.

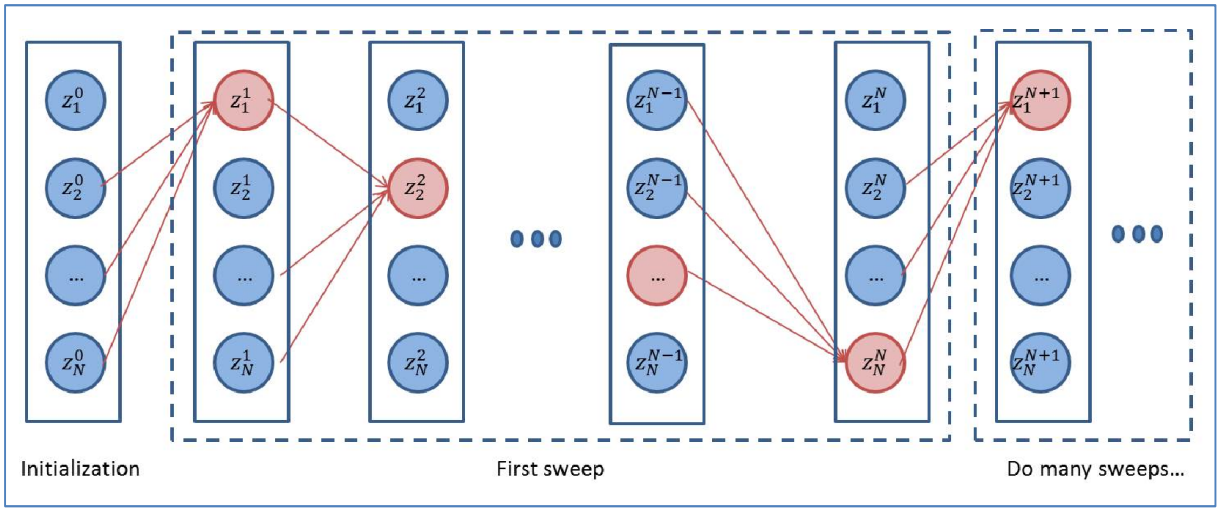
Gibbs sampling

The Gibbs sampling algorithm starts from a vector z^0 initialized randomly and then updates it component by component. We refer to an **iteration over all components** as a **Gibbs sweep** (if z is N-dimensional, the samples obtained after Gibbs sweeps will be z^N, z^{2N}, z^{3N}, ….) .

The update of a component z_i consists in resampling from the conditional distribution of this component, conditioned on all other components:

$$z_i^{s+1} \sim p\left(z_i \middle| z_{\setminus i}^s\right)$$

The figure display Gibbs sampling updates of z (grouped by sweeps with dashed line).



Gibbs sampling for LDA

For LDA, we can derive the conditional distribution for the i[th] token (from the expression above)

$$p\left(z_i \middle| z_{\setminus i}^s, w, \alpha, \beta\right) \propto \frac{\alpha + C_{d_i\, z_i}^{D-i}}{K\alpha + \sum_{k=1}^{K} C_{d_i\, k}^{D-i}} \times \frac{\beta + C_{w_i\, z_i}^{W-i}}{V\beta + \sum_{w=1}^{V} C_{w\, z_i}^{W-i}}$$

where we define the reduced counts matrices:

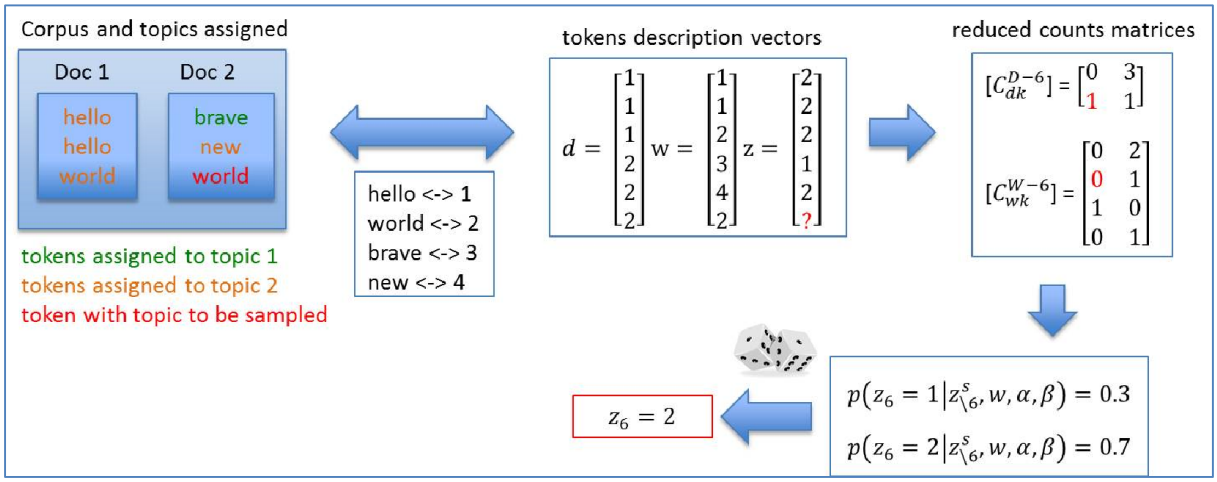
- $C_{dk}^{D-i} = \#\{j \in \{1,\dots,N\}\setminus i \colon d_j = d \text{ and } z_j^s = k\}$ is the number of times topic k is assigned to document d, without counting token i.
- $C_{wk}^{W-i} = \#\{j \in \{1,\dots,N\}\setminus i \colon w_j = w \text{ and } z_j^s = k\}$ is the number of times topic k is assigned to word w, without counting token i.

Loosely speaking, this conditional distribution tells us that sampling topic z_i for the i[th] token is more likely:

- if many tokens in the documents d_i have been assigned topic z_i
- if many tokens corresponding to the word w_i have been assigned topic z_i

Appendix A gives the detailed algorithm to perform the sampling.

The figure below shows one step of sampling for the previous example: we resample token 6, while keeping fix the topics assigned to other tokens. We remove the token 6 from counts matrices, and compute probabilities of sampling topic 1 or topic 2 (e.g. 0.3 or 0.7 respectively). We then draw the topic of token 6 randomly, according to these probabilities, for example we sample topic 2!



After enough sampling steps, we obtain a sample vector z^s which should represent a 'likely' configuration of the LDA model. There is absolutely no theoretical guarantee for that, but the fact that we observed this sample in such a high dimensional space makes it likely that we are *not too far from a local mode of the distribution*.

Estimation of θ and ϕ

The section above explains how to obtain samples of z. From these, we can obtain corresponding estimates of θ and ϕ, which are usually the quantities of interest.

Basic estimation based on one sample

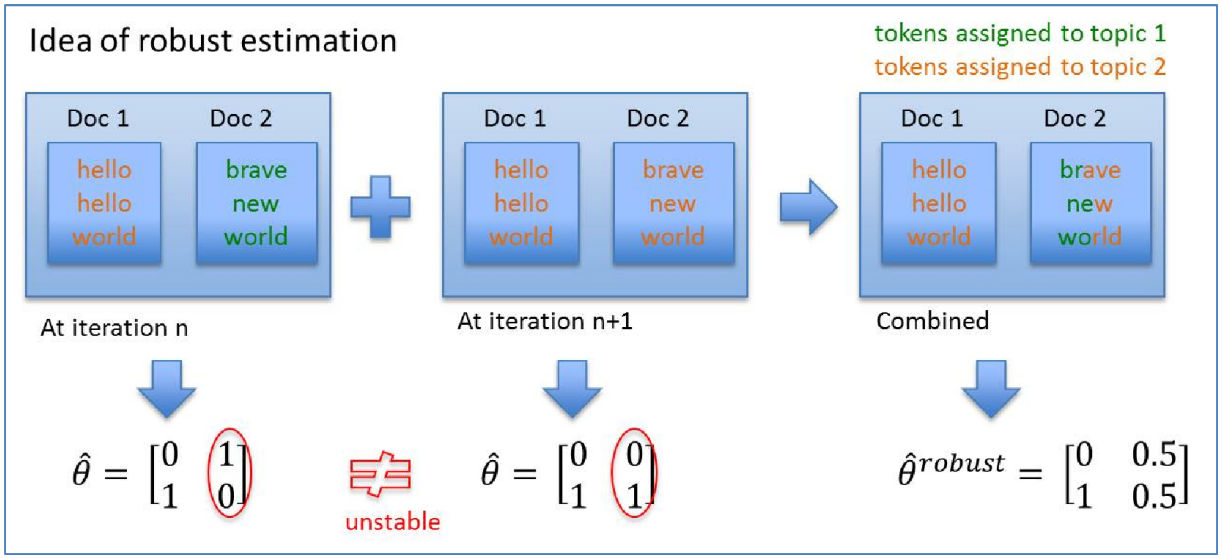
By the conjugate property of multinomial and Dirichlet, we have that $p(\theta_d|z,\alpha)$ is a K-dimensional Dirichlet distribution: $\theta_d \sim Dir(\alpha + C_d^D.)$. Therefore, if we are given the topics assigned z, the MLE estimator of θ_d is the mode of the Dirichlet distribution, i.e. $\hat{\theta}_{kd}^{MLE} = \frac{\alpha + C_{dk}^D - 1}{K\alpha + \sum_{j=1}^K C_{dj}^D - K}$. However this mode only exists if $\alpha + C_{dk}^D > 1 \; \forall k$, which is rarely the case, therefore the quantity usually used is the mean $\hat{\theta}_{kd} = \frac{\alpha + C_{dk}^D}{K\alpha + \sum_{j=1}^K C_{dj}^D}$: .

Similarly, we have $\phi_k \sim Dir(\beta + C_{.k}^W)$ and we usually use the estimator $\hat{\phi}_{wk} = \frac{\beta + C_{wk}^W}{V\beta + \sum_{j=1}^V C_{jk}^W}$.

Robust estimation

The basic estimation is based on a single sample, which can cause $\hat{\theta}$ and $\hat{\phi}$ to vary a significantly between samples. In particular, *$\hat{\theta}$ tends to be unstable when there are few tokens per document.*

To solve this problem, we first build a robust estimate $\hat{\phi}^{robust}$, which we then use to obtain a robust estimate $\hat{\theta}^{robust}$. By **robust**, we mean that the estimators shouldn't change too much based on a single topics assignment. For example, if a document d has only three words, which can belong to two different topics, then an estimator $\hat{\theta}$ based on only one sample will change drastically across samples. As we cannot know which assignment is the correct one, we would prefer an estimator expressing this uncertainty.



To build a robust estimate $\hat{\phi}^{robust}$, we combine several samples, obtained by performing one Gibbs sweep starting from a stable sample. The obtained samples are highly correlated, and this is what we want, because we want to stay around the same local mode (avoid 'switching and melding topics'), while making our estimator robust.

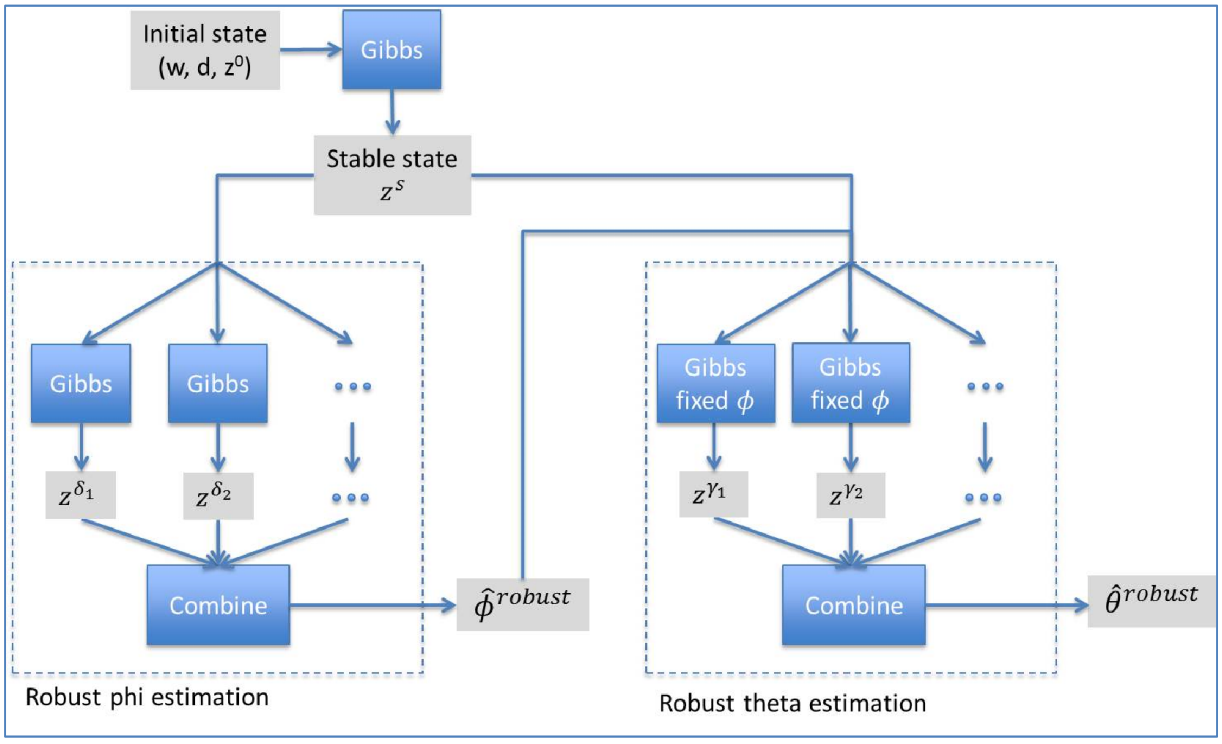
To combine these samples and build a robust $\hat{\phi}^{robust}$, we first average C_{wk}^{W} over the samples to obtain $\bar{C}_{wk}^{W}$. We then use the same estimator form as before, i.e. $\hat{\phi}_{wk}^{robust} = \frac{\beta + \bar{C}_{wk}^{W}}{V\beta + \sum_{j=1}^{V} \bar{C}_{jk}^{W}}$.

We then build a robust estimate $\hat{\theta}^{robust}$ by keeping $\hat{\phi}^{robust}$ fixed and combining samples for this topic structure. We perform Gibbs sampling as before, but using the following equation:

$$p\left(z_i \middle| z_{\setminus i}^{s}, w, \alpha, \hat{\phi}^{robust}\right) \propto \frac{\alpha + C_{d_i z_i}^{D-i}}{K\alpha + \sum_{k=1}^{K} C_{d_i k}^{D-i}} \times \hat{\phi}_{w_i z_i}^{robust}$$

As the topic structure is now fixed, we can combine many samples (obtained from different initializations) without risks of switching topics. To combine the samples and build a robust $\hat{\theta}^{robust}$, we first average C_{dk}^{D} over the samples to obtain $\bar{C}_{dk}^{D}$. We then use the same estimator form as before, i.e. $\hat{\theta}_{kd}^{robust} = \frac{\alpha + \bar{C}_{dk}^{D}}{K\alpha + \sum_{j=1}^{K} \bar{C}_{dj}^{D}}$.

To summarize, using this robust estimation procedure, we obtain two (corresponding) estimators $\hat{\phi}^{robust}$ and $\hat{\theta}^{robust}$, which should be robust to sampling artifacts. The figure below summarizes the process.



Automatic parameters fitting

We developed an automatic method to find good values for hyper-parameters α and β and use them during model fitting. This approach takes hyper-parameters which maximize the likelihood of the fitted model, and allows saving time searching for hyper-parameters. Note that it can still be interesting to use other values in order to force sparser topics or sparser documents.

The approach used is to maximize the likelihood of the model for the hyper-parameter given the observed counts. For α, it corresponds to the maximization of

$$L(\alpha) = \prod_{d=1}^{D} p\left([C_{d\cdot}^{D}]|\alpha\right)$$

with $[C_{d\cdot}^{D}] \sim Mult(\theta_d)$ and $\theta_d \sim Diriclet(\alpha)$. This leads to find the root of the derivative of the log-likelihood:

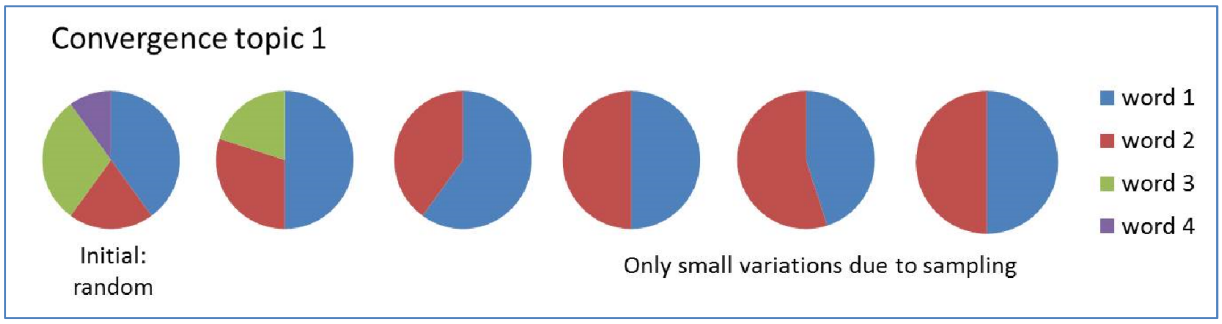
$$l'(\alpha) = DK\psi(K\alpha) - DK\psi(\alpha) + \sum_{d=1}^{D}\sum_{k=1}^{k}\psi\left(\alpha + C_{dk}^{D}\right) + K\sum_{d=1}^{D}\psi\left(K\alpha + \sum_{k=1}^{K} C_{dk}^{D}\right) = 0$$

where ψ is the digamma function. We find the root using bisection method in a reasonable range for α.

A similar approach is used to find hyper-parameter beta.

Convergence criterion

An issue with LDA is that it can be hard to tell whether the model fitted is close to a mode, local at least. A reasonable and easy-to-compute indicator is the stability of the topics ϕ. When we are far from a mode, the topics tend to change quickly, whereas when we are close from a mode, the topics tend to be stable and their variations are due to sampling randomness. The figure below illustrates this phenomenon for a given topic.



Therefore our convergence criterion computes the distance between consecutive topics ϕ as

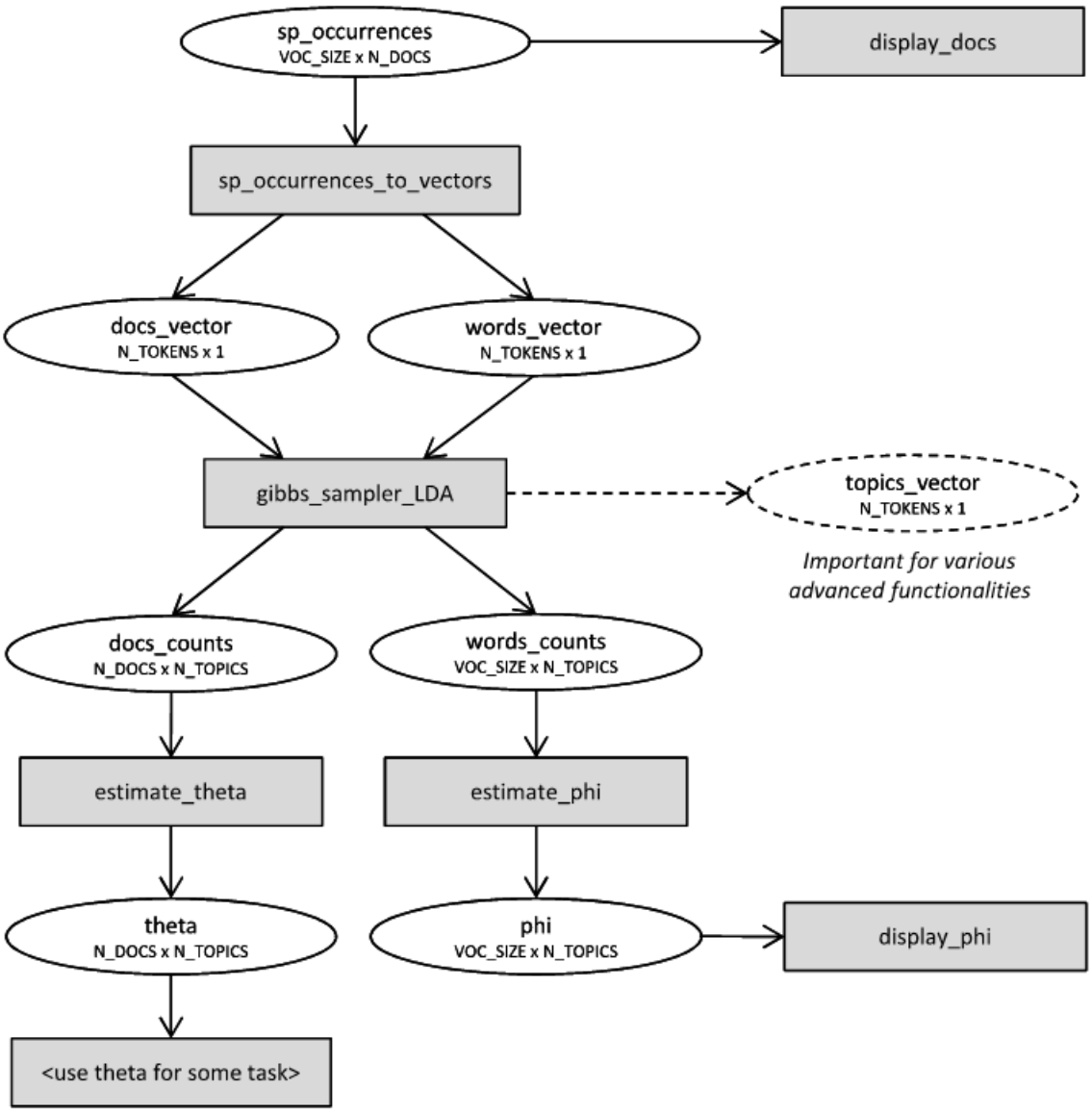
$$dist\left(\phi^{old}, \phi^{new}\right) = \frac{1}{2K}\sum_{k=1}^{K}\sum_{w=1}^{V} abs(\phi_{wk}^{old} - \phi_{wk}^{new}) \in [0,1]$$

and average it over a given number of samples. This (averaged) distance corresponds to the variability of the topics and decreases until it stabilizes and oscillates around some low value, at which point we consider that convergence is reached.

LDA toolbox variable naming

Math. representation	Variable name	Type
N	N_TOKENS	scalar
K	N_TOPICS	scalar
D	N_DOCS	scalar
V	VOC_SIZE	scalar
w	words_vector	Nx1 vector
d	docs_vector	Nx1 vector
z	topics_vector	Nx1 vector
$[\phi_{wk}]$	phi	VOC_SIZE xN_TOPICS matrix
$[\theta_{kd}]$	theta	N_TOPICSxN_DOCS matrix
$[C_{wk}^{W}]$	words_counts	VOC_SIZE xN_TOPICS sparse matrix
$[C_{dk}^{D}]$	docs_counts	N_DOCSxN_TOPICS sparse matrix

LDA toolbox basic workflow



Appendix A: Detailed Gibbs sampling algorithm

As $p\left(z_i \middle| z_{\setminus i}^s, w, \alpha, \beta\right)$ is a probability distribution, it must sum to one over possible values of z_i. Therefore we can further simplify (and compute the normalizing factor at the end):

$$p\left(z_i \middle| z_{\setminus i}^s, w, \alpha, \beta\right) \propto \left(\alpha + C_{d_i\, z_i}^{D-i}\right) \times \frac{\beta + C_{w_i\, z_i}^{W-i}}{V\beta + \sum_{w=1}^{V} C_{w\, z_i}^{W-i}} \equiv q_i^s(z_i)$$

The algorithm then runs as follow:

1) *Initialization*: Initialize z^0 randomly
2) *Gibbs sweep*: For token i = 1 to N
    a. Obtain matrices $[C_{dk}^{D-i}]$ and $[C_{wk}^{W-i}]$ from $[C_{dk}^{D}]$ and $[C_{wk}^{W}]$
    b. Compute $q_i(k)$ for k = 1 to K
    c. Draw a random number U uniformly between 0 and $\sum_{k=1}^{K} q_i^s(k)$
    d. Pick topic z_i^{s+1} such that $\sum_{k=1}^{z_i^{s+1}} q_i^s(k) \leq U < \sum_{k=1}^{z_i^{s+1}+1} q_i^s(k)$
    e. Obtain matrices $[C_{dk}^{D}]$ and $[C_{wk}^{W}]$ from $[C_{dk}^{D-i}]$ and $[C_{wk}^{W-i}]$ with new topic z_i^{s+1}
3) *Repeating*: Go back to step 2 until some criterion is met (e.g. fix number of sweep)