Summary of Agustinus's Gibbs Sampler

Thomas W. Jones

7/25/2018

This is a summary of a blog post on LDA Gibbs sampling and a quick R implmentation of it. The post and Python code can be found here: https://wiseodd.github.io/techblog/2017/09/07/lda-gibbs/. I have also saved a PDF copy of the page as I found it on 7/25/2018 in /readings/02_lda/other_ref/.

Defining the model

LDA has, of course, the same generative process regardless of author. Where Agustinus differs is that he formally declares the posterior. The process for this is generally to define the model and derive the posterior as Geigle does. However, Geigle does not state the posterior so explicitly. And Geigle's formulation may be different as he integrates out some parameters. I am not sure if that's what Agustinus does either. More algebra is required!

The original model using (approximately) Agustinus's notation:

$$\boldsymbol{\pi}_i \sim Dirichlet(\boldsymbol{\alpha}) \tag{1}$$
$$\mathbf{z}_{i,v} \sim Categorical(\boldsymbol{\pi}_i) \tag{2}$$
$$\mathbf{b}_k \sim Dirichlet(\boldsymbol{\gamma}) \tag{3}$$
$$\mathbf{y}_{i,v} \sim Categorical(z_{i,v} = k, \mathbf{B}) \tag{4}$$

In the above, I assume that $\mathbf{B}$ is a matrix whose k rows consist of $\mathbf{b}_k$. (Analagous to ϕ_k and $\boldsymbol{\Phi}$ in other sources.) And I assume that the notation $z_{i,v} = k, \mathbf{B}$ means the k-th row of $\mathbf{B}$, depending on which was sampled at the previous step.

The posterior:

$$P(z_{i,v} = k|\boldsymbol{\pi}_i, \mathbf{b}_k) \propto exp\{\ln(\pi_{i,k}) + \ln(b_{k,y_{i,v}})\} \tag{5}$$
$$P(\boldsymbol{\pi}_i|z_{i,v} = k, \mathbf{b}_k) = Dirichlet(\alpha + \sum_l \mathbb{I}(z_{i,l} = k)) \tag{6}$$
$$P(\mathbf{b}_k|z_{i,v} = k, \pi_i) = Dirichlet(\gamma + \sum_i \sum_l \mathbb{I}(y_{i,l} = v, z_{i,l} = k)) \tag{7}$$

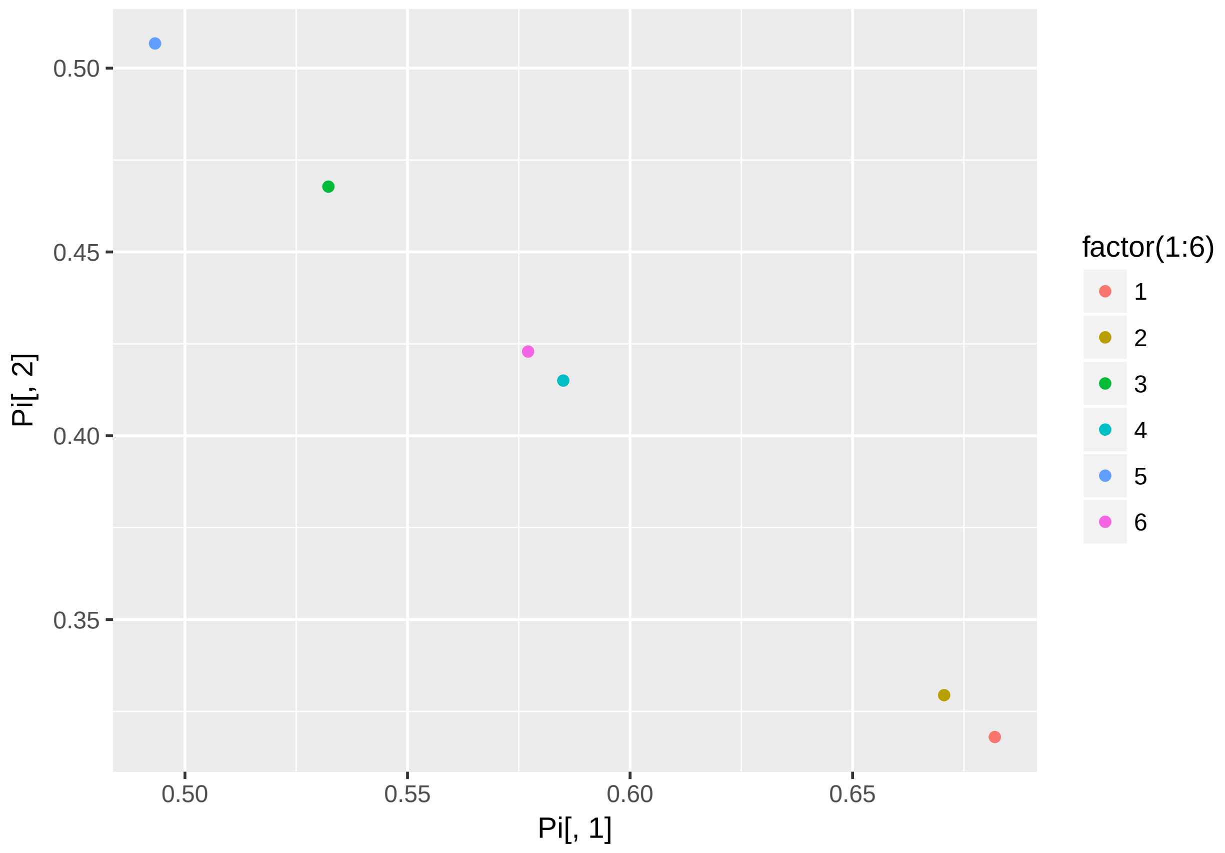
Implemented code

The code on Agustinus's blog is in Python. I've done my best to implement it in R below. As of this writing (07/29/2018), the result is consistent, but not the same as what Agustinus gets. His result is that documents $\{1, 2, 3\}$ should be high in the same topic and $\{4, 5, 6\}$ should be high in the other one. What I get (regardless of random seed) is that documents $\{1, 2, 4\}$ are high in the same topic, $\{3, 6\}$ are high in the other, and $\{5\}$ is split almost evenly.

Again, as of this writing, I need to go through this code in more depth to make sure that (a) it is doing the same calculations as Agustinus and (b) it is doing what I think it should. (Note that these may or may not be the same thing.)

```r
### Declare some toy data ----
# document term matrix of word counts
X <- rbind(c(0, 0, 1, 2, 2),
           c(0, 0, 1, 1, 1),
           c(0, 1, 2, 2, 2),
           c(4, 4, 4, 4, 4),
           c(3, 3, 4, 4, 4),
           c(3, 4, 4, 4, 4))

W <- seq_len(ncol(X))

N_d = nrow(X)  # num of docs
N_w = length(W)  # num of words
N_k = 2  # num of topics

### Initialize parameters ----
# set.seed(8675309)

alpha <- rep(1, N_k)
gamma <- rep(1, N_w) # I usually call this "beta"

# Z := word topic assignment, initialized with random integers
# This would have to be a triplet or other sparse matrix
Z <- matrix(sample.int(10, N_w * N_d, replace = TRUE), ncol = N_w, nrow = N_d)

# Pi := document topic distribution
Pi <- gtools::rdirichlet(N_d, alpha)

# B := word topic distribution
B <- gtools::rdirichlet(N_k, gamma)

### Here is the Gibbs sampler ----

num_iter <- 1000

for (j in 1:num_iter) { # for each iteration

  # Sample from full conditional of Z
  for (i in seq_len(N_d)) { # For each document
    for (v in seq_len(N_w)) { # for each word
      # Calculate params for Z
      p_iv <- exp(log(Pi[i,]) + log(B[,v]))

      # p_iv <- p_iv / sum(p_iv) # normalization not required in R

      # Resample word topic assignment Z
      Z[i,v] <- which.max(rmultinom(1, 1, p_iv))
    }
  }

  # Sample from the full conditional of Pi
  for (i in seq_len(N_d)) { # for each document
    for (v in seq_len(N_w)) { # for each word
```

```r
      m <- numeric(N_k)

      # gather sufficient statistics
      for (k in seq_len(N_k)) {
        m[k] <- sum(Z[i,] == k)
      }

      # resample doc-topic distribution
      Pi[i,] <- gtools::rdirichlet(1, alpha + m)
    }
  }

  # Sample from the full conditional of B
  for (k in seq_len(N_k)) {
    n <- numeric(N_w)

    # Gather sufficient statistics
    for (v in seq_len(N_w)) {
      for (i in seq_len(N_d)) {
        for (l in seq_len(N_w)) {
          n[v] <- n[v] + (X[i, l] == v & Z[i,l] == k)
        }
      }
    }

    # resample word topic distribution
    B[k,] <- gtools::rdirichlet(1, gamma + n)
  }

}

ggplot2::qplot(x = Pi[,1], y = Pi[,2], colour = factor(1:6))
```


0.50
0.45
0.40
0.35
Pi[, 2]
0.50
0.55
0.60
0.65
Pi[, 1]
factor(1:6)
1
2
3
4
5
6