Simulation Using pLDA

Penalized Latent Dirichlet Allocation (pLDA) is an adaption of LDA to single-cell RNA sequencing data, where the gene expression profiles can be inferred. This PDF demonstrates the use of pLDA package using a simulated dataset.

Simulation Setup

In this simulated dataset, we set the number of topics to be 5, the number of cells to be 20, and the number of genes to be 100.

The topic-gene matrix (β matrix) is divided into three blocks: 30 genes have high variation across each topic, 30 genes have medium variation across each topic, and 40 genes are housekeeping genes which have the same expression level.

First, we simulate the frequency for 100 genes from an exponential distribution with $\lambda = 100$ so that each gene has an expected frequency of 0.01. Within each block, the genes are ordered from highest frequency to lowest frequency. The resulting vector of frequencies is replicated 5 times for each topic.

Next, we generate random variation among different topics. For the highly varied genes, we add a noise term $exp(N(0,1))$ to the frequency, while the standard error is reduced to 0.5 for mediumly varied genes. For the housekeeping genes, there is no noise term. Finally, we normalize the simulated frequencies so that they add up to 1 for each topic.

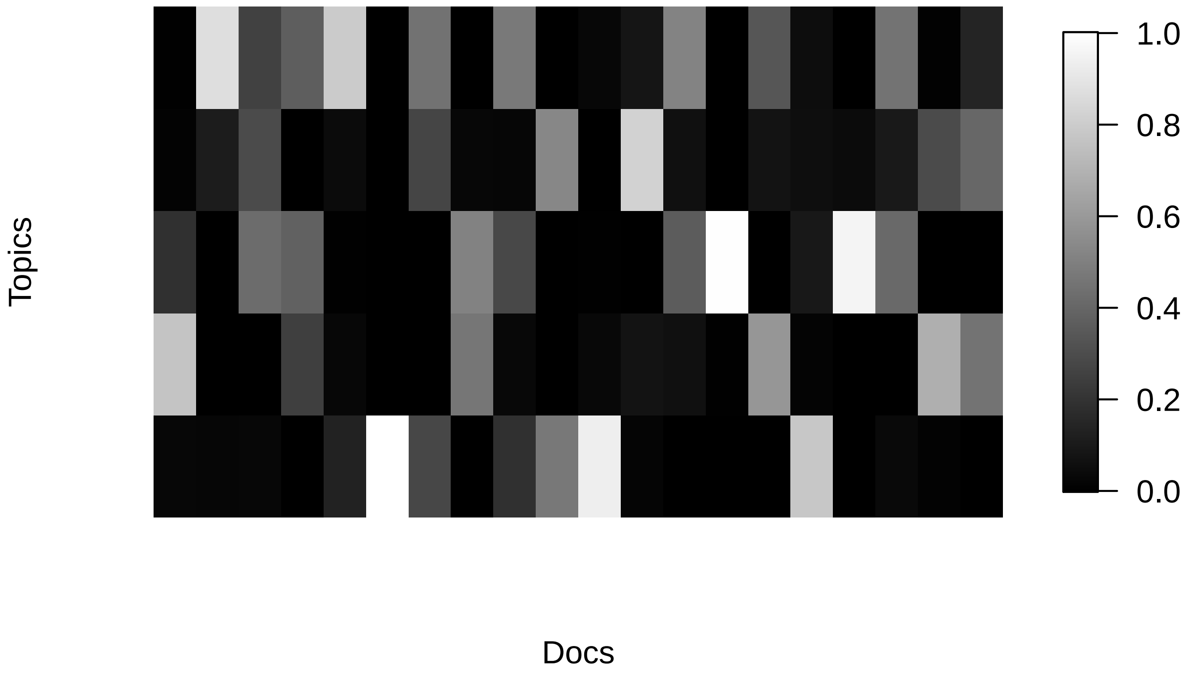
```r
library(gtools)
library(pLDA)
library(fields)
# simulation setup
set.seed(2018)
n_topics <- k <- 5
n_docs <- 20
n_vocab <- V <- 100
n_noiseGene <- 5

# beta matrix
# 5 rows, 100 columns
beta <- matrix(rep(rexp(100, V), k), nrow = k, ncol = V, byrow = T)
# rearrange according to signal
beta[, 1:30] <- beta[, order(beta[1, 1:30], decreasing = T)]
beta[, 31:60] <- beta[, order(beta[1, 31:60], decreasing = T) + 30]
beta[, 61:100] <- beta[, order(beta[1, 61:100], decreasing = T) + 60]
# add noise
# gene 1:30: high variation
beta[, 1:30] <- beta[, 1:30] * exp(rnorm(k * 30, 0, 1))
# gene 31:60: medium variation
beta[, 31:60] <- beta[, 31:60] * exp(rnorm(k * 30, 0, 0.5))
# gene 61:100: hk, no variation


# rowSums(beta) should be equal to 1
for (i in 1:k) {
  beta[i, 1:60] <- beta[i, 1:60] * (1 - sum(beta[i, 61:100])) / sum(beta[i, 1:60])
}
beta.true <- beta
```

The θ matrix represents the topic-document matrix. It is generated from a Dirichlet distribution. Here, the Dirichlet prior is chosen to be 0.2 for all topics. The simulated θ matrix can be viewed in the image below.

```
# theta mat
theta.true <- rdirichlet(n_docs, rep(1/n_topics, n_topics))
plot.theta <- function(theta) {
  image.plot(t(theta)[, nrow(theta):1],
             col = grey(seq(0, 1, length = 256)),
             axes = F,
             zlim = c(0, 1),
             ylab = 'Topics',
             xlab = 'Docs')
}
plot.theta(t(theta.true))
```



The word count in the document is then simulated using the β and θ matrices from a multinomial distribution.

```
docs <- matrix(0, nrow = n_docs, ncol = n_vocab)
ksai <- 1000000 # average words per doc
for (i in 1:n_docs) {
  # draw topics for each word
  tops <- rmultinom(1, rpois(1, ksai), theta.true[i, ])
  # draw words
  for (j in 1:n_topics) {
    docs[i, ] <- docs[i, ] + rmultinom(1, tops[j], beta.true[j, ])
  }
}
```

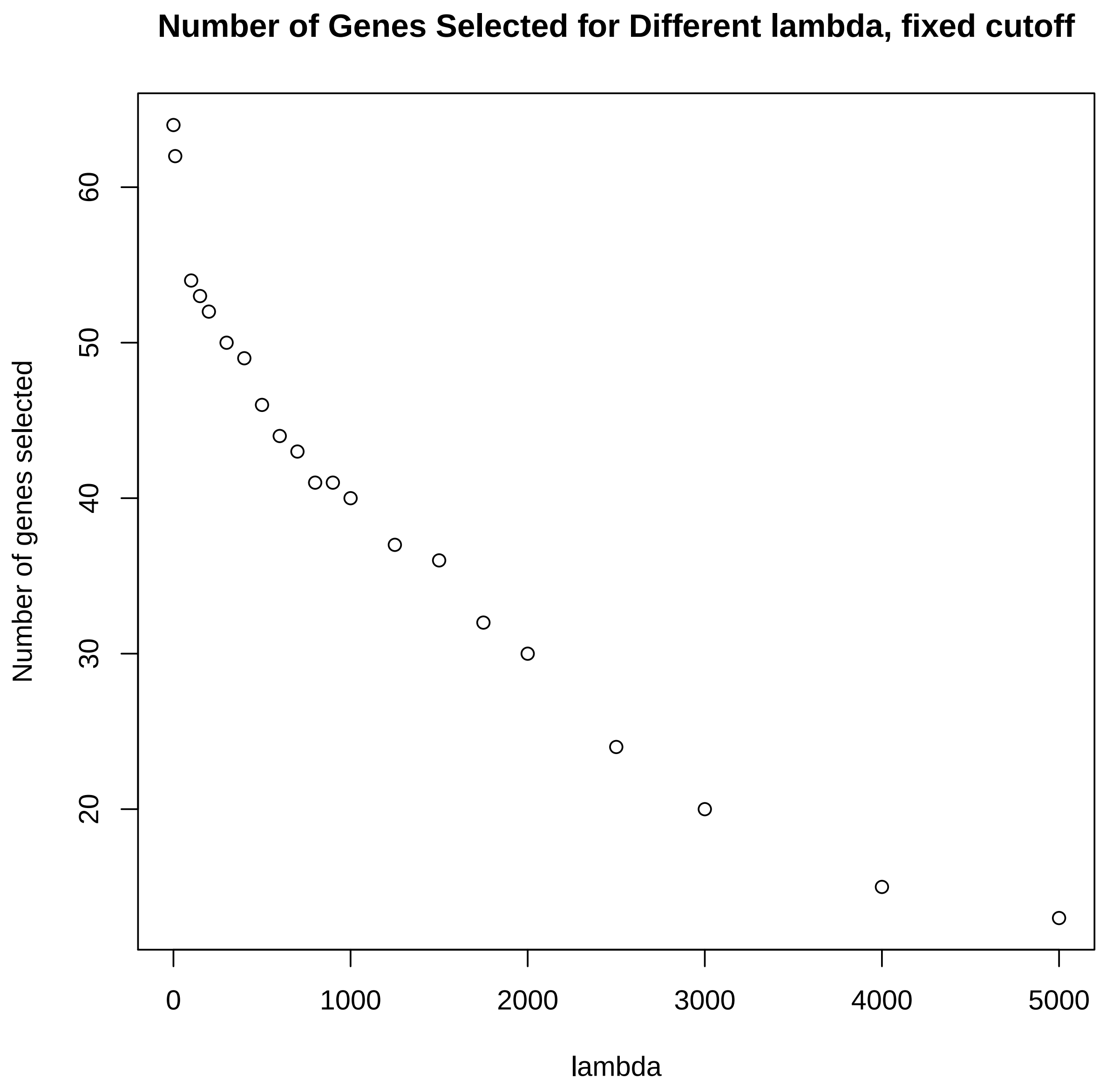
Running the Package

To run the pLDA model, we first need to choose the shrinkage parameter λ, using the plda_choose_lambda function. The input arguments include a vector of different λ values, and the indices of housekeeping genes.

```
tmp <- plda_choose_lambda(dat = t(docs),
                          k = 5,
                          lam_values = c(0, 10, 100, 150, 200, 300, 400, 500,
                                         600, 700, 800, 900, 1000, 1250, 1500,
                                         1750, 2000, 2500, 3000, 4000, 5000),
                          idx_hk = 61:100,
                          fdir = "./output/")
```

```
## [1] "Calculating for 5 topics."
## [1] "Processing lambda=0..."
## [1] "Processing lambda=10..."
## [1] "Processing lambda=100..."
## [1] "Processing lambda=150..."
## [1] "Processing lambda=200..."
## [1] "Processing lambda=300..."
## [1] "Processing lambda=400..."
## [1] "Processing lambda=500..."
## [1] "Processing lambda=600..."
## [1] "Processing lambda=700..."
## [1] "Processing lambda=800..."
## [1] "Processing lambda=900..."
## [1] "Processing lambda=1000..."
## [1] "Processing lambda=1250..."
## [1] "Processing lambda=1500..."
## [1] "Processing lambda=1750..."
## [1] "Processing lambda=2000..."
## [1] "Processing lambda=2500..."
## [1] "Processing lambda=3000..."
## [1] "Processing lambda=4000..."
## [1] "Processing lambda=5000..."
```

This function returns the number of interesting genes for different values of λ, whose variance is greater than the average variance of housekeeping genes in the estimated β matrix. λ is chosen at the elbow point of the curve.



Next, we run the pLDA model with the chosen λ, using the plda function.

```
fit800 = plda(docs, k = n_topics, lambda = 800)
beta.plda800 = exp(fit800$logProbW)
row.rearrange800 = greedyRearrangeColumn(t(beta.true), t(beta.plda800))
beta.plda800 = beta.plda800[row.rearrange800, ]
```

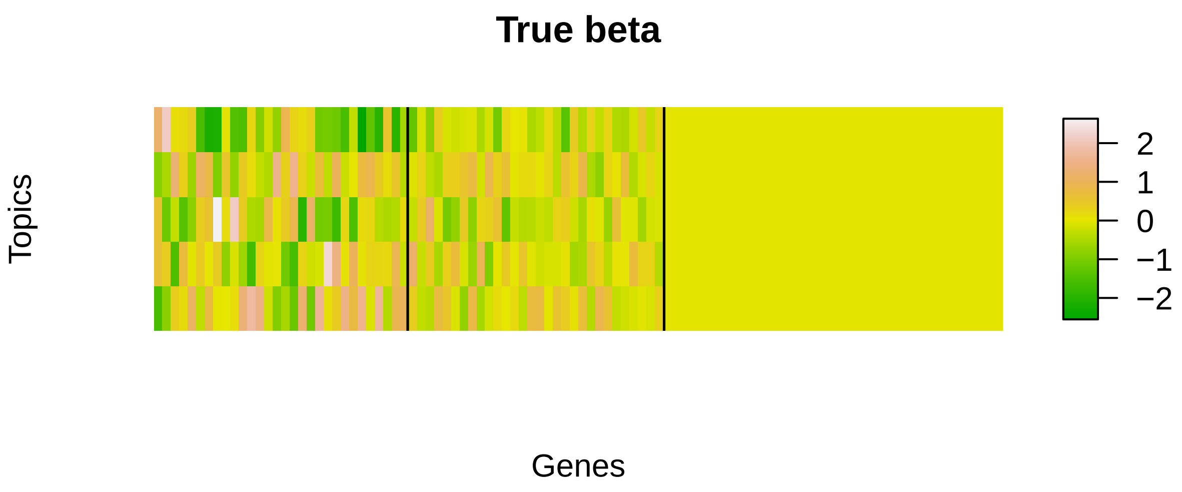
The output contains several objects. logProbW is the estimated beta matrix on the log scale, and gamma is the estimated θ (γ) matrix.

The returned β matrix does not have the same row order as the original true β matrix used for simulation. Therefore we need to convert it back to the same order using the `greedyRearrangeColumn` function.

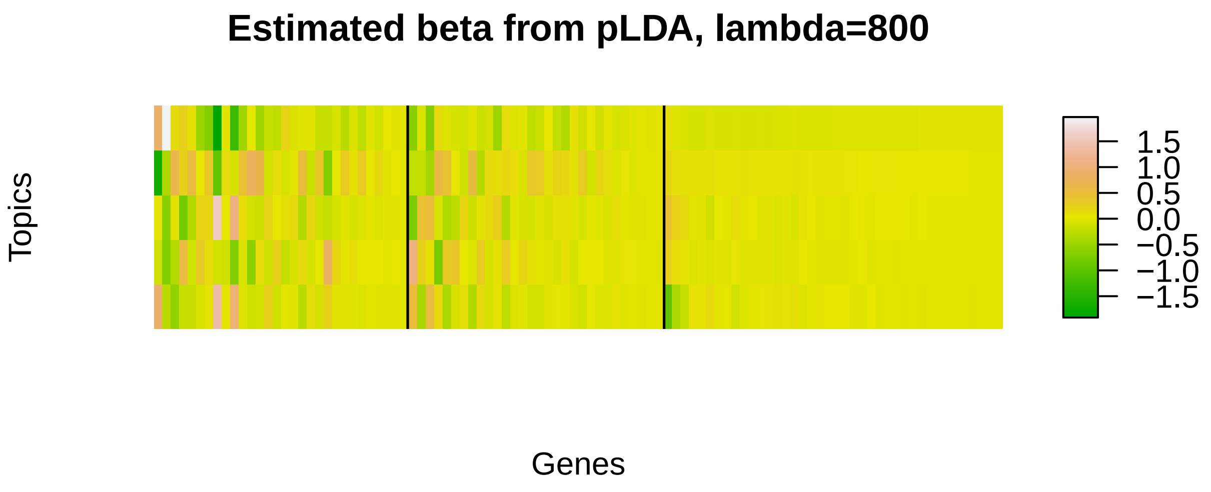
Comparing the beta.plda and beta.true:

```r
# normalize beta so that can visualize noise signal
normalize <- function(beta){
  beta <- log(beta)
  return(sweep(beta, 2, colMeans(beta)))
}

# plot beta matrix
plot.beta = function(beta){
  image.plot(t(normalize(beta))[, nrow(normalize(beta)):1],
             col = terrain.colors(256),
             axes = F,
             ylab = 'Topics',
             xlab = 'Genes')
  segments(0.297, -0.2, 0.297, 1.2, lwd = 1.3)
  segments(0.602, -0.2, 0.602, 1.2, lwd = 1.3)
}
{
  plot.beta(beta.true)
  title("True beta")
}
```



```r
{
    plot.beta(beta.plda800)
    title("Estimated beta from pLDA, lambda=800")
}
```



Most of the selected genes are the highly-varied genes. If λ is further increased, more non-interesting genes will be eliminated.