BST249 HW4

Sijia Huo

3/17/2022

1.

The target distribution can be expanded as $p\left(z, w, \theta \mid x_{\text{obs}}\right) \propto \prod_{i=1}^{n}\left(w_{z_i} \prod_{\ell=1}^{L} \prod_{c=1}^{2} \prod_{k=1}^{K} \prod_{v=1}^{V_\ell} \theta_{k\ell}(v)^{\mathrm{I}(z_i=k)\mathrm{I}(x_{i\ell c}=v)o_{ilc}}\right)$
Then we have

(i)

$$
\begin{aligned}
p\left(z_i \mid w, \theta, x_{\text{obs}}\right) &\propto w_{z_i} \prod_{k=1}^{K} \prod_{\ell=1}^{L} \prod_{c=1}^{2} \prod_{v=1}^{V_\ell} \theta_{k\ell}(v)^{\mathrm{I}(z_i=k)\mathrm{I}(x_{i\ell c}=v)o_{ilc}} \\
&\propto \prod_{k=1}^{K}\left\{w_k^{\mathrm{I}(z_i=k)} \prod_{\ell=1}^{L} \prod_{c=1}^{2} \prod_{v=1}^{V_\ell} \theta_{k\ell}(v)^{\mathrm{I}(z_i=k)\mathrm{I}(x_{i\ell c}=v)o_{ilc}}\right\} \\
&\propto \prod_{k=1}^{K}\left\{w_k \prod_{\ell=1}^{L} \prod_{c=1}^{2} \prod_{v=1}^{V_\ell}\left(\theta_{k\ell}(v)^{\mathrm{I}(x_{i\ell c}=v)o_{ilc}}\right)\right\}^{\mathrm{I}(z_i=k)} \\
&= \prod_{k=1}^{K}\left\{w_k \prod_{\ell=1}^{L} \prod_{c=1}^{2}\left(\theta_{k\ell}(x_{ilc})^{o_{ilc}}\right)\right\}^{\mathrm{I}(z_i=k)} \\
&\propto \text{Categorical } (\omega)
\end{aligned}
$$

where $\omega = (\omega_1, \omega_2, ...\omega_K)$ and $\omega_k = \frac{w_k \prod_{\ell=1}^{L} \prod_{c=1}^{2} \theta_{k\ell}(x_{ilc})^{o_{ilc}}}{\sum_{m=1}^{K} w_m \prod_{\ell=1}^{L} \prod_{c=1}^{2} \theta_{m\ell}(x_{ilc})^{o_{ilc}}}$

(ii)

$$
\begin{aligned}
p\left(w \mid z, \theta, x_{\text{obs}}\right) &\propto \prod_{i=1}^{n} w_{z_i} \\
&\propto \prod_{k=1}^{K} w_k^{\sum_{i=1}^{n} \mathrm{I}(z_i=k)} \\
&\propto \text{Dirichlet } (\alpha)
\end{aligned}
$$

where $\alpha = (\alpha_1, \alpha_2, ...\alpha_K)$ and $\alpha_k = \sum_{i=1}^{n} \mathrm{I}\left(z_i = k\right) + 1$

(iii)

$$
\begin{aligned}
p\left(\theta_{kl} \mid w, z, x_{\text{obs}}\right) &\propto \prod_{v=1}^{V_\ell}\left\{\prod_{i=1}^{n}\prod_{c=1}^{2}\theta_{k\ell}(v)^{\mathrm{I}(z_i=k)\mathrm{I}(x_{i\ell c}=v)o_{ilc}}\right\} \\
&\propto \prod_{v=1}^{V_\ell}\left\{\theta_{k\ell}(v)^{\sum_{i=1}^{n}\sum_{c=1}^{2}\mathrm{I}(z_i=k)\mathrm{I}(x_{i\ell c}=v)o_{ilc}}\right\} \\
&\propto \text{Dirichlet }(\lambda)
\end{aligned}
$$

where $\lambda = (\lambda_1, \lambda_2, ...\lambda_{V_\ell})$ and $\lambda_v = \sum_{i=1}^{n}\sum_{c=1}^{2}\mathrm{I}\left(z_i = k\right)\mathrm{I}\left(x_{i\ell c} = v\right)o_{ilc} + 1$

2.

Since assuming a uniform prior, we first randomly sample $Z_i \sim \text{Categorical}(1/K, ..., 1/K)$ for all i. However, this uniform proir may also cause the issue of label switching. The rest of the code is displayed as below.

```
library(gtools)
library(ggplot2)

############## helper functions ##############

# calculate weight for certain k when sampling z for sample i
# not multiplying by w_k
z_sampler_prob_k_i= function(theta_k_cur,x_i){
  ret = 1 # assuming no sample has completely missing values
  for(i in 1:length(x_i)){
    if(x_i[i]>0) ret = ret*theta_k_cur[ceiling(i/2),x_i[i]]
  }
  return(ret)
}

# sample z for sample i, return the sampling probability
z_sampler_prob_i= function(theta_cur,x_i,w_cur){
  # numerator of the omega in categorical distribution
  omega_k_num = w_cur * sapply(theta_cur,z_sampler_prob_k_i,x_i=x_i)
  return((omega_k_num/sum(omega_k_num)))
}

# sample w
w_sampler = function(z_cur,K){
  alpha = sapply(1:K, function(x) 1+sum(z_cur == x))
  return(rdirichlet(1,alpha))
}

# sample theta_k for a specific k
theta_sampler_k = function(k,x,V,z) {
  ret = matrix(0,nrow=length(V),ncol=max(V))
  # filter out the samples with z_i=k
  x_k = x[which(z == k),]
  if(sum(z == k) == 1) x_k = t(as.matrix(x_k)) # edge case 1, preserve dimension

  for (i in 1:length(V)){
    cur_lambda = rep(0,V[i])
    if(sum(z == k) == 0){
```

```
      cur_lambda = rep(1,V[i])
    } else{
      for(j in 1:V[i]){
        cur_lambda[j] = sum(x_k[,c(2*i-1,2*i)] == j)+1
        }
    }
    ret[i,1:V[i]] = rdirichlet(1,cur_lambda) # sample theta_kl
  }
  return(ret)
}

############## main function ##############

# n interation of sampling. K components,each locus has length V
gibbs_sampler_n = function(K,V,x,round){

  # construct data structure to hold results for each iteration
  n_obs = nrow(x)
  z_iter = matrix(0,nrow = round+1, ncol = n_obs) # z for each round
  w_iter = matrix(0,nrow = round, ncol = K) # w for each round
  theta_iter = vector("list", round) # theta for each round

  # initialization, z's starting point
  z_iter[1,] = sample(1:K,size = n_obs, replace = TRUE, prob = rep(1/K,K))

  # start simulation
  for (i in 1:round){
    w_iter[i,]= w_sampler(z_iter[i,],K) #sample w
    theta_iter[[i]] = lapply(1:K,theta_sampler_k,x=x,V=V,z=z_iter[i,]) #sample theta
    #sample z, first get probability for each obs, one row each obs, then sample
    z_prob = t(apply(x,1,z_sampler_prob_i,theta_cur=theta_iter[[i]],w_cur=w_iter[i,]))
    if(K==1) z_prob = t(z_prob) # if k=1, reformat the data structure
    z_iter[i+1,] = apply(z_prob,1,function(x) sample(1:K,size = 1, replace = TRUE, prob = x))
  }

  # return lists and the last iteration of z_prob
  return(list(z_iter[-(round+1),],w_iter,theta_iter,z_prob))
}
```

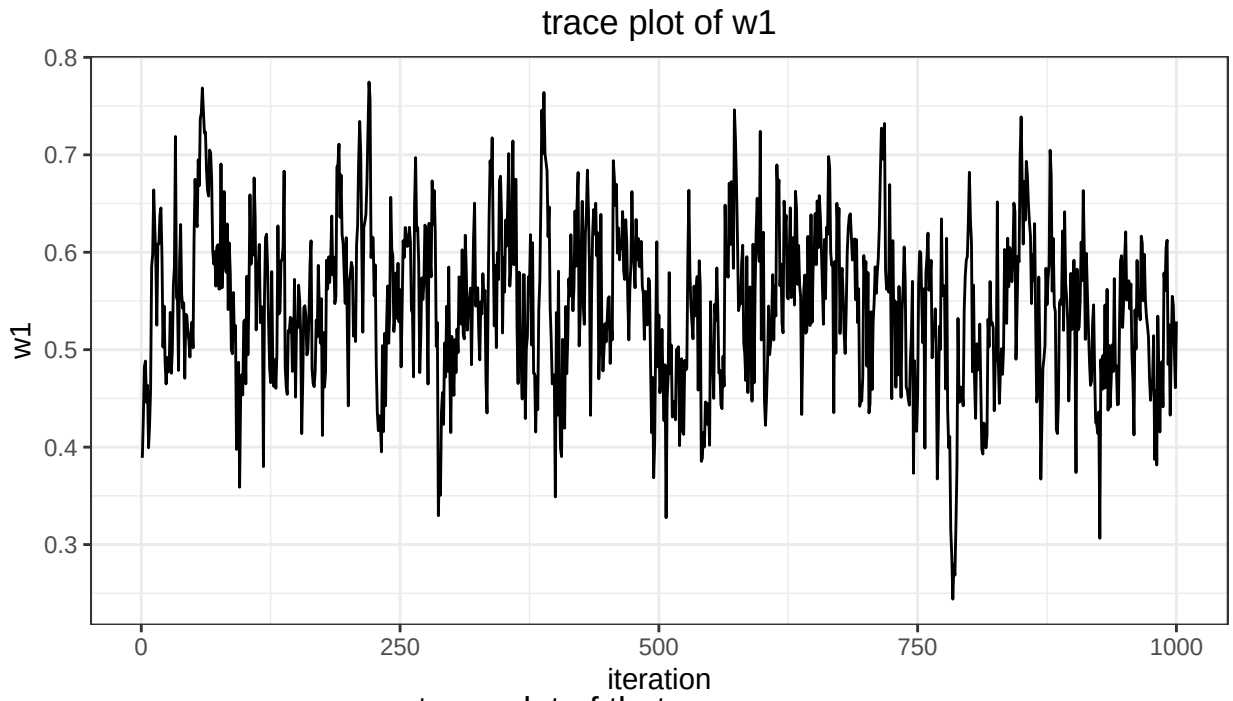
3.

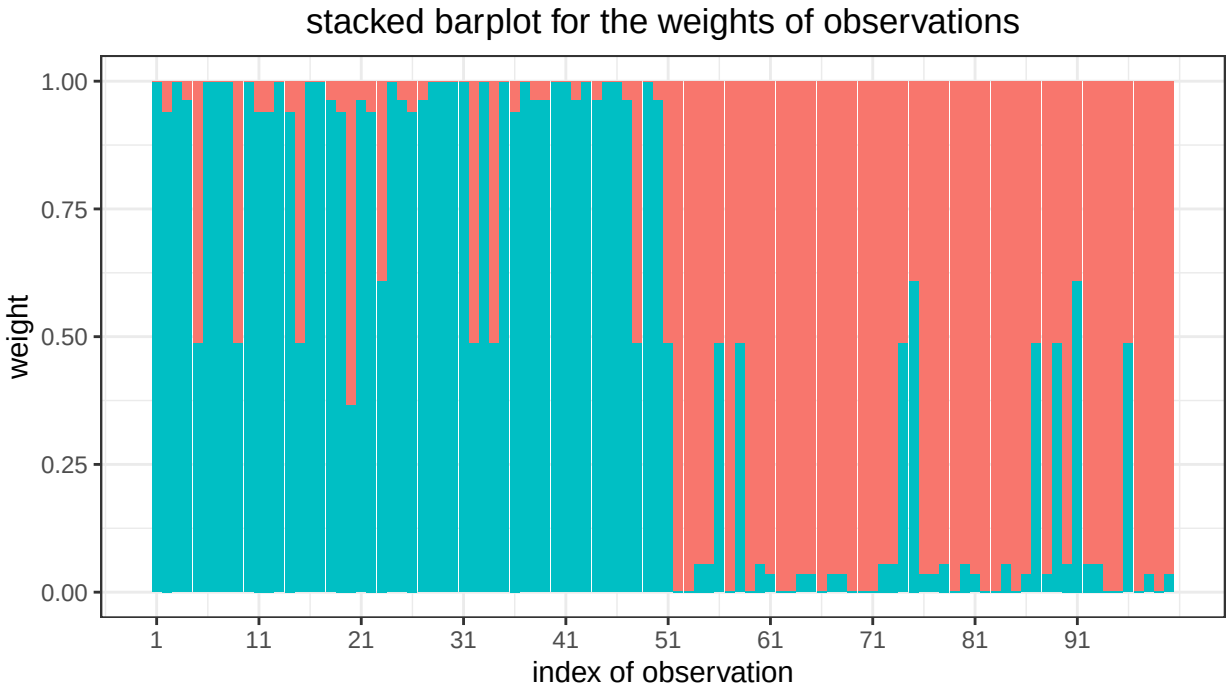
```
# generate simulated data

sim_data_generator = function(q,n){
  set.seed(0329)
  z1 = replicate(n/2,{sample(1:2,size = 4, replace = TRUE, prob = c(q,1-q))}) # generate for z=1,p(v=1)
  z2 = replicate(n/2,{sample(1:2,size = 4, replace = TRUE, prob = c(1-q,q))}) # generate for z=2,p(v=1)
  return(t(cbind(z1,z2)))
}
```

(a)

```r
# a function to do plotting

plot_maker = function(result_list,trace = FALSE, stack_bar = TRUE){

  if(trace){
    # trace for w
    df_trace_w = data.frame(cbind(w1 = result_list[[2]][,1], iter = 1:nrow(result_list[[2]])))
    trace_w = ggplot(df_trace_w,aes(iter)) +
      geom_line(aes(y = w1)) +
      ggtitle("trace plot of w1") +
      theme_bw() +
      theme(plot.title = element_text(hjust = 0.5)) +
      xlab("iteration") + ylab("w1") + theme(aspect.ratio=3/6)
    print(trace_w)

    # trace for theta
    theta_list = result_list[[3]]

    df_trace_theta = data.frame(theta_11 = sapply(theta_list,"[[",1)[1,], theta_12 = sapply(theta_list,
                                theta_21 = sapply(theta_list,"[[",2)[1,], theta_22 = sapply(theta_list,
                                iter = 1:length(theta_list))

    trace_theta = ggplot(df_trace_theta,aes(iter)) +
      geom_line(aes(y = theta_11, colour = "theta11(1)")) +
      geom_line(aes(y = theta_12, colour = "theta12(1)")) +
      geom_line(aes(y = theta_21, colour = "theta21(1)")) +
      geom_line(aes(y = theta_22, colour = "theta22(1)")) +
      ggtitle("trace plot of theta") +
      theme_bw() +
      theme(plot.title = element_text(hjust = 0.5)) +
      xlab("iteration") + ylab("theta") + theme(aspect.ratio=3/6)
    print(trace_theta)
  }

  if(stack_bar){

    prob_z = result_list[[4]]
    df_stack_w = data.frame(prob = c(t(prob_z)), obs_ind = rep(1:nrow(prob_z),each = ncol(prob_z)),
                            component = as.character(rep(1:ncol(prob_z)),times = nrow(prob_z)))
    stacked_z = ggplot(df_stack_w, aes(fill = component, y=prob, x=obs_ind)) +
      scale_x_continuous(breaks = seq(1,nrow(prob_z),by = 10)) +
      geom_bar(position="fill", stat="identity") +
      ggtitle("stacked barplot for the weights of observations") +
      theme_bw() +
      theme(plot.title = element_text(hjust = 0.5), legend.position="none") +
      xlab("index of observation") + ylab("weight") + theme(aspect.ratio=3/6)
    print(stacked_z)
  }
}

# run the simulation, sample, plot

data_3_a = sim_data_generator(0.8,100)
```

```
ret_3_a = gibbs_sampler_n(2,c(2,2),data_3_a,1000)
plot_maker(ret_3_a, TRUE, TRUE)
```



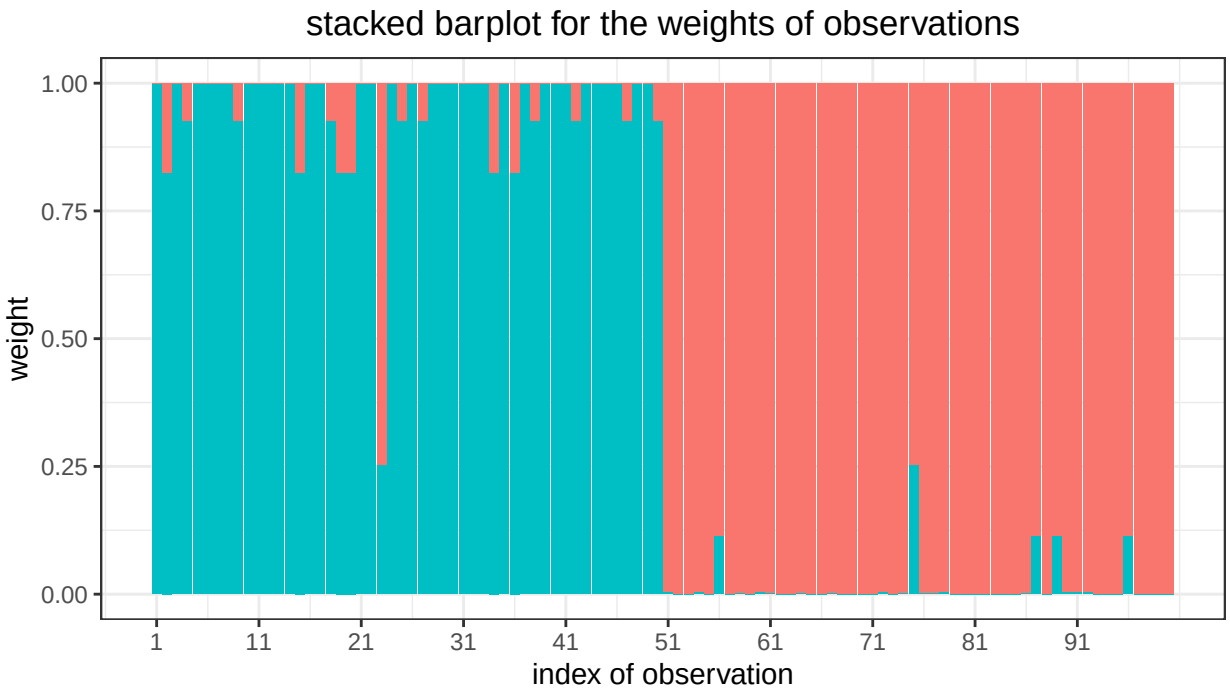


The MCMC sampler appear to be doing reasonably well. For the trace plot of w, we hope that the mean of the value to be around 0.5 for there are equal amount of observations in two populations, and this seems to be true in our plot. For the trace plot of $\theta_{kl}(1)$, we hope that there's no significant difference between two loci. In addition, for population 1, the probability that variant equals to 1 is about 0.8 whereas for population 2, the probability equals to 0.2. Indeed, in our plot, $\theta_{11}(1), \theta_{12}(1)$ overlap with each other and both have mean at around 0.8; $\theta_{21}(1), \theta_{22}(1)$ overlap with each other with both mean at around 0.2. Finally, the stacked barplot shows that the two populations can be seperated comparatively well (most of the first 50 observations are classified as population 1 and most of the rests are grouped into population 2) even though the noise still exists.

(b)

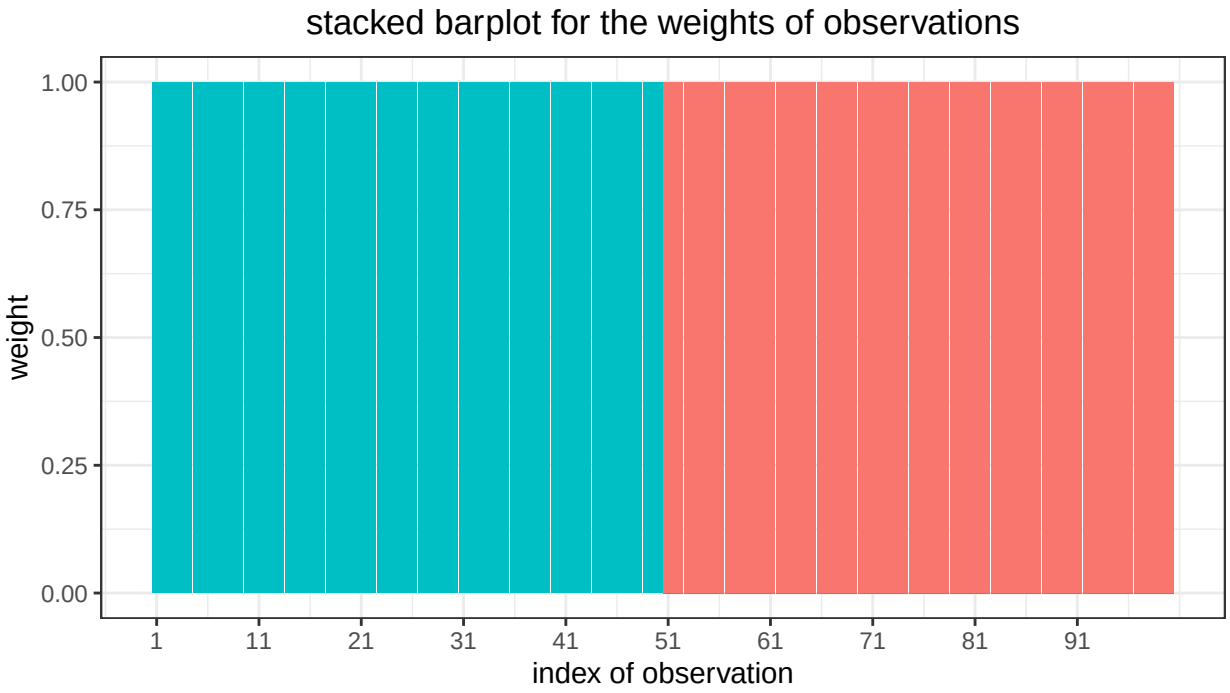
for q = 0.9

```
# q = 0.9

data_3_b1 = sim_data_generator(0.9,100)
ret_3_b1 = gibbs_sampler_n(2,c(2,2),data_3_b1,1000)
plot_maker(ret_3_b1, FALSE, TRUE)
```



for q = 0.99

```
# q = 0.99

data_3_b2 = sim_data_generator(0.99,100)
ret_3_b2 = gibbs_sampler_n(2,c(2,2),data_3_b2,1000)
plot_maker(ret_3_b2, FALSE, TRUE)
```

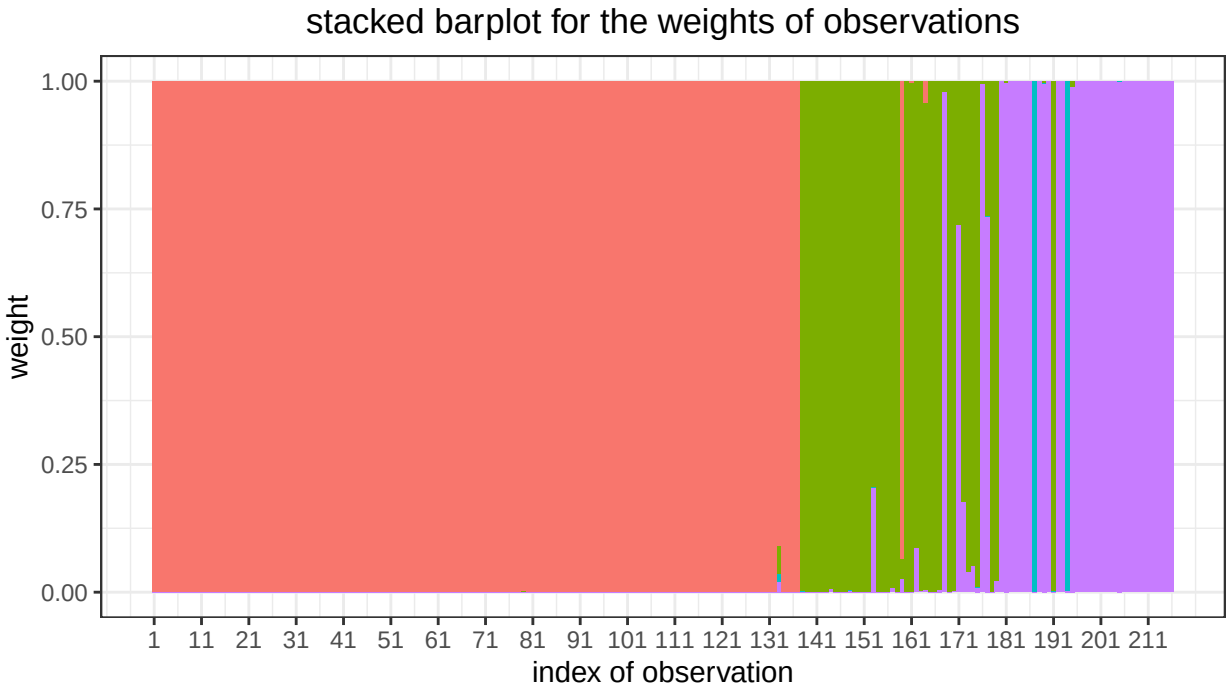


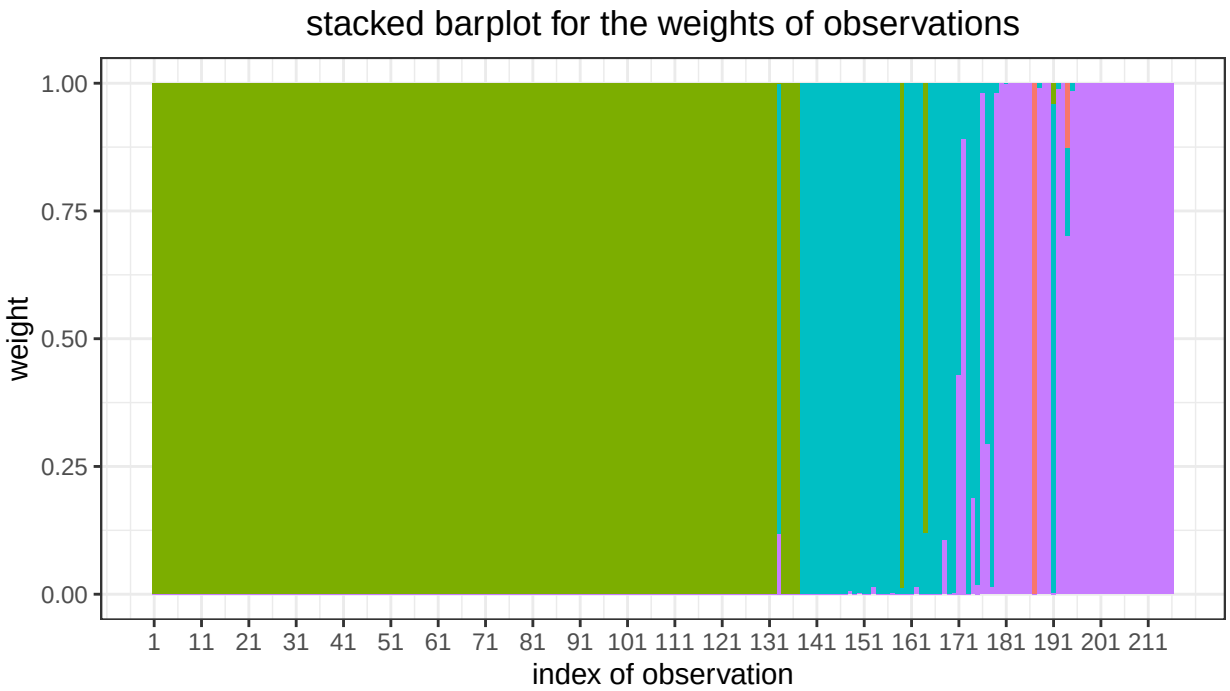
The result does make sense. As q increases (from 0.8 to 0.9 and then to 0.99), the difference of the distribution of loci become more and more significant between the two populations and therefore it becomes easier to easier

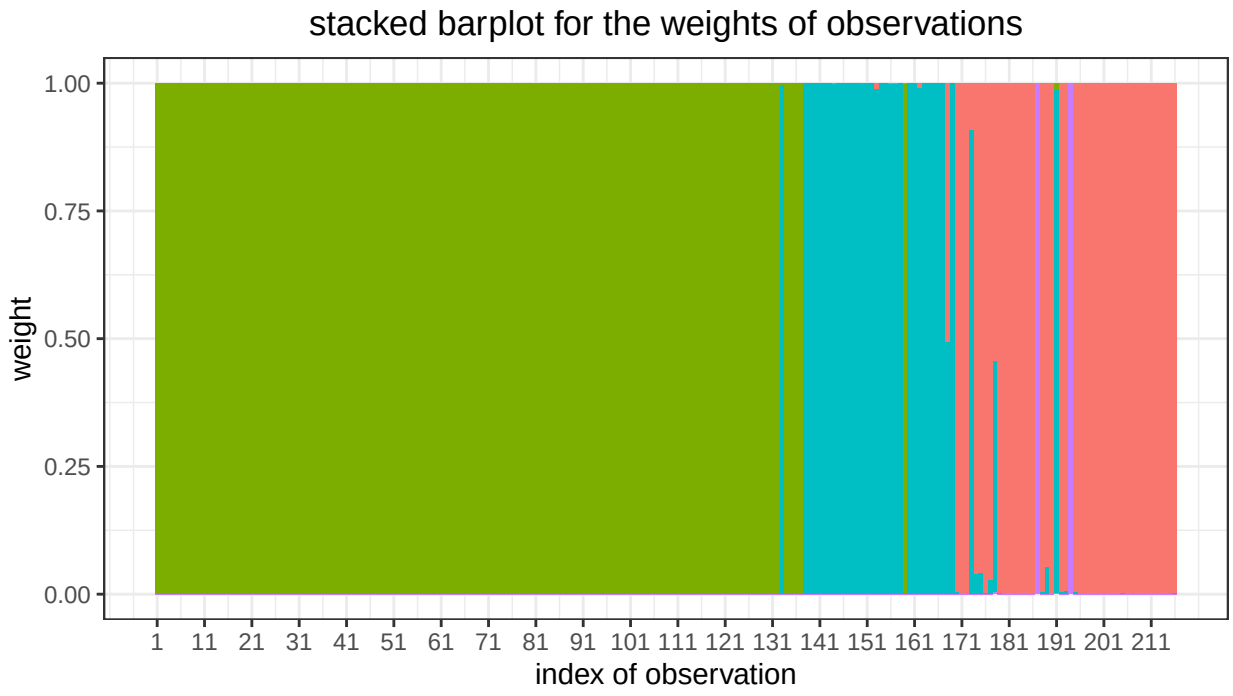
to be corretly classified. And indeed, when $q = 0.99$, the sampler can correctly classify all the observations.

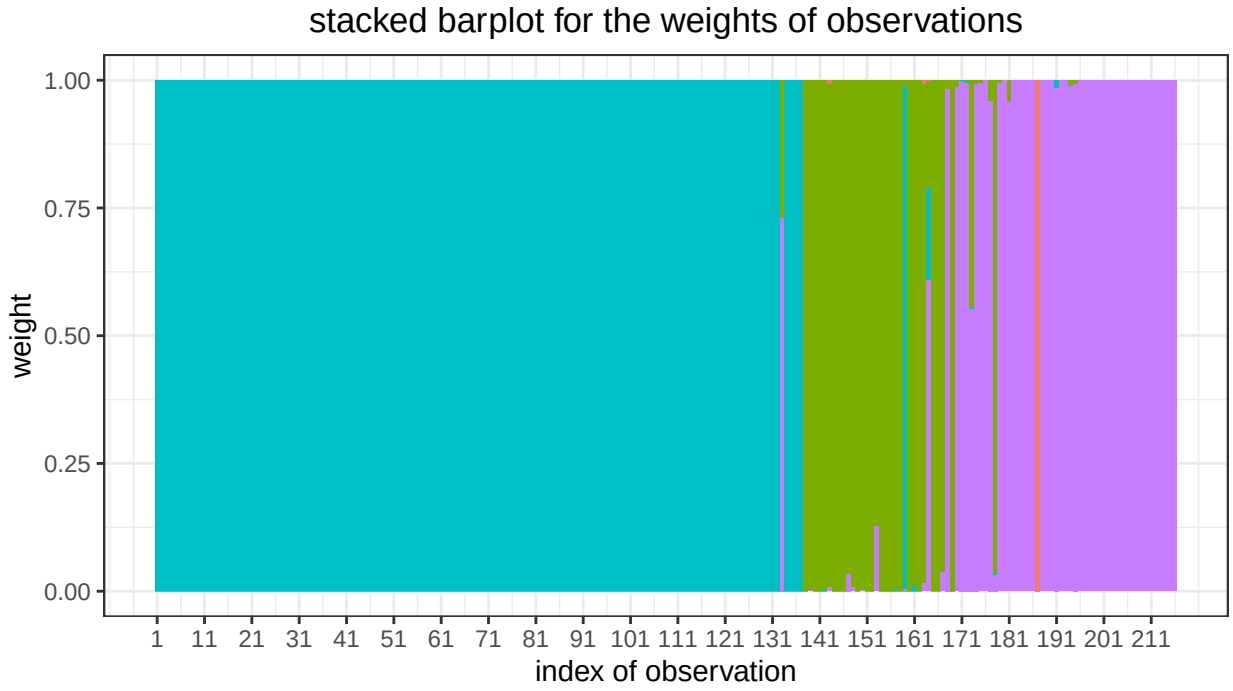
4.

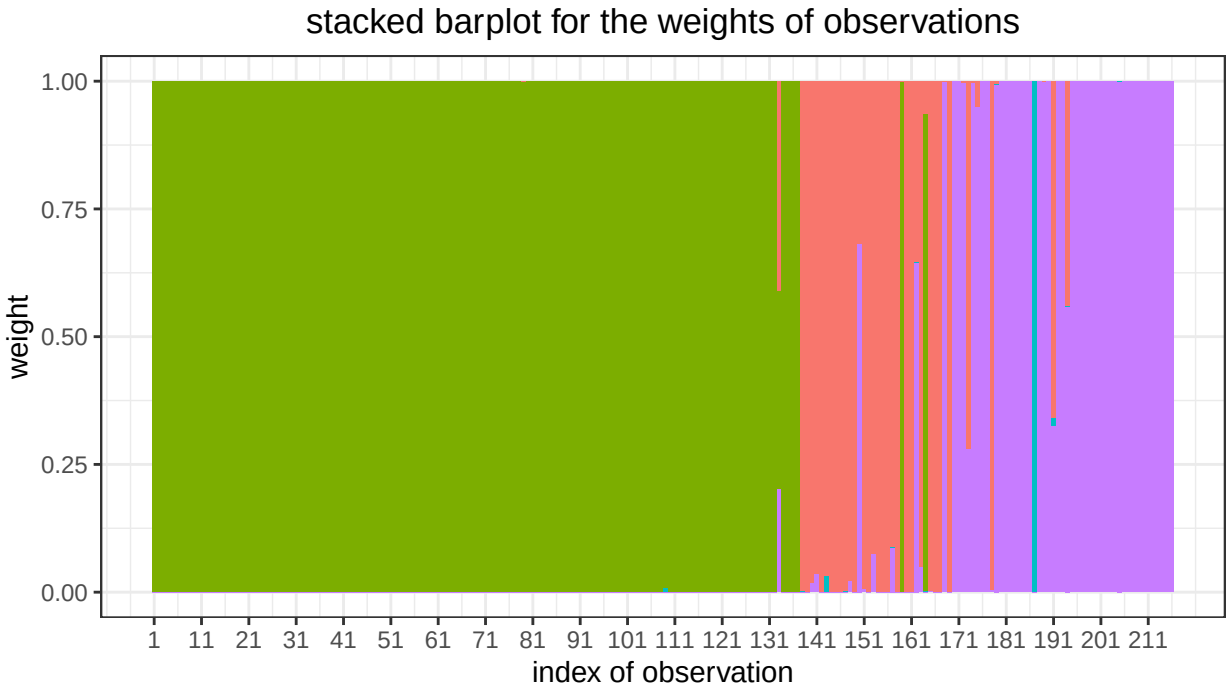
```
data <- read.table("~/Documents/Spring2022/BST249/Homework/HW4/homework-4-data.txt", quote="\"", commen
data = as.matrix(data)
V = c(15,13,6,6,9,14,16,9)
ret_4 = replicate(5,gibbs_sampler_n(K=4,V=V,x=data,round=1000),FALSE)
invisible(lapply(ret_4,plot_maker)) # print out the plots
```






stacked barplot for the weights of observations
weight
1.00
0.75
0.50
0.25
0.00
1 11 21 31 41 51 61 71 81 91 101 111 121 131 141 151 161 171 181 191 201 211
index of observation


stacked barplot for the weights of observations
weight
1.00
0.75
0.50
0.25
0.00
1 11 21 31 41 51 61 71 81 91 101 111 121 131 141 151 161 171 181 191 201 211
index of observation



First, we can observe three blocks with different colors in all five plots. This means that our classifications are consistent among the replicates. And generally speaking, there might be three mixture components among the observed population (1-137,138-169,170-216). Besides, the colors labeled are different from one to another, indicating the issue of label switching.

5.

My results do appear to roughly make sense. All the replicates (rougly) classify observation 1-137 into one group and the rest of observatios into other two groups. This matches the classification bewteen "common impala" and "black-faced impala". It can also be observed that the black-faced impala from CH, SH, KAF and LU (index 138-169) are roughly grouped together each time whereas the the rest observations (black-faced impala from SE, BU, IM, SA) are classified into another group. This indicates that our sampler can (roughly) correctly group the observations from the same regions together.

The two animals that are misclassified most of the time are No.133 and No.159. No. 133 has 8 missing values and No. 159 has 6 missing values. The incomplete information hinders the correct classification. Moreover, if the observations have certain variant that is more common in another subspecies than in its own subspecies, this may also cause the mis-classification.

6.

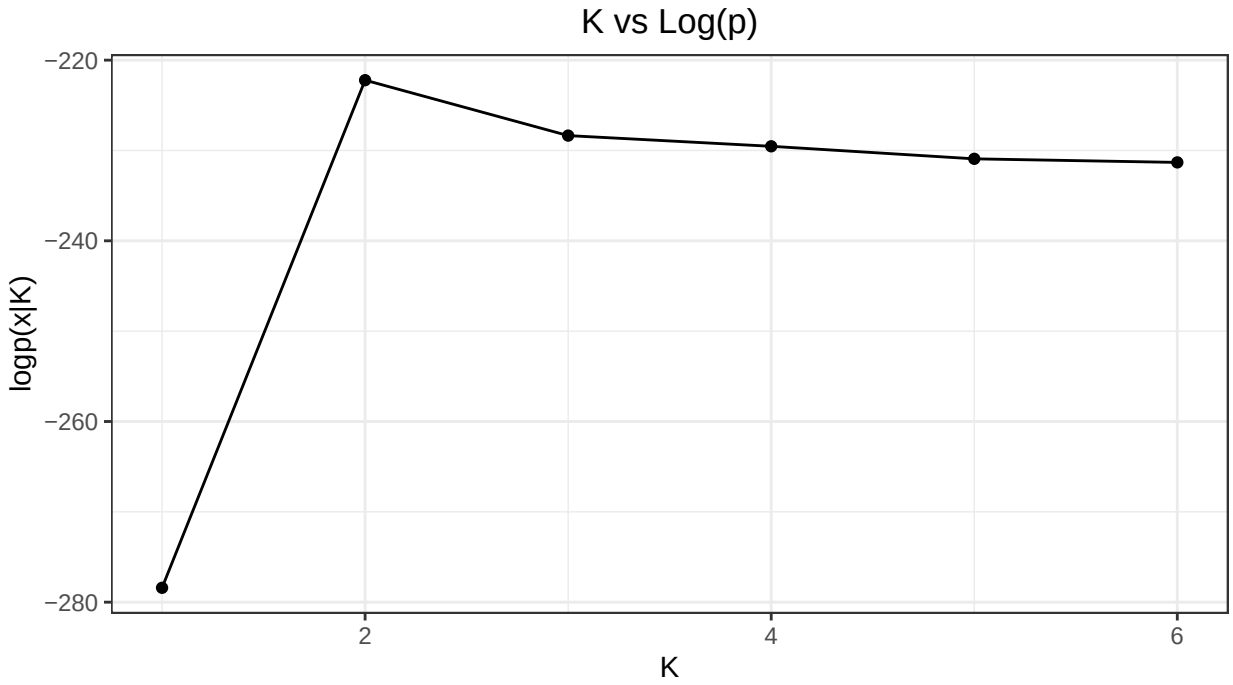
$$
\begin{aligned}
\log p\left(x_{\text{obs}} \mid z^{(t)}, w^{(t)}, \theta^{(t)}\right) &= \log \prod_{i}^{n} p\left(x_{\text{i}} \mid z^{(t)}, w^{(t)}, \theta^{(t)}\right) \\
&= \log \prod_{k=1}^{K} \prod_{l=1}^{L} \prod_{v=1}^{V_l} \theta_{kl}^{(t)}(v)^{\sum_{i=1}^{n} \sum_{c=1}^{2} \mathrm{I}\left(z_i^{(t)}=k\right) \mathrm{I}(x_{i\ell c}=v) o_{ilc}} \\
&= \sum_{k=1}^{K} \sum_{l=1}^{L} \sum_{v=1}^{V_l} \log\{\theta_{kl}^{(t)}(v)^{\sum_{i=1}^{n} \sum_{c=1}^{2} \mathrm{I}\left(z_i^{(t)}=k\right) \mathrm{I}(x_{i\ell c}=v) o_{ilc}}\}
\end{aligned}
$$

Based on the derivation, implement the method as follows.

```
############# helper functions #############

# calculate sum(log(x_i| z,theta)) for time t and all observations i whose z_i=k
logp_k_t = function(x,k,t,V,result_list){
  z  = result_list[[1]][t,] # z of t's round
  if(sum(z == k) == 0) return(0)
  x_k = x[which(z == k),]
  if(sum(z == k) == 1) x_k = t(as.matrix(x_k)) # preserve dimension
  theta_k = result_list[[3]][[t]][[k]]
  ret = 0
  for (i in 1:length(V)){
    for(j in 1:V[i]){
      ret = ret+(sum(x_k[,c(2*i-1,2*i)] == j))*log(theta_k[i,j])
    }
  }
  return(ret)
}

# calculate sum(log(x_i| z,theta)) for all k and all observations
logp_t = function(x,t,V,K,result_list){
  return(sum(sapply(1:K,logp_k_t,x=x,t=t,V=V,result_list=result_list)))
}

############# main function #############

# calcualte log(P(x|K))
logp_x = function(x,K,V,result_list){
  log_p_x = sapply(1001:6000,logp_t,x=x,V=V,K=K,result_list=result_list)
  mu_hat = mean(-2 * log_p_x)
  sigma2_hat = mean((-2*log_p_x - mu_hat)^2)
  return((-mu_hat/2 - sigma2_hat/8))
}
```

(a)

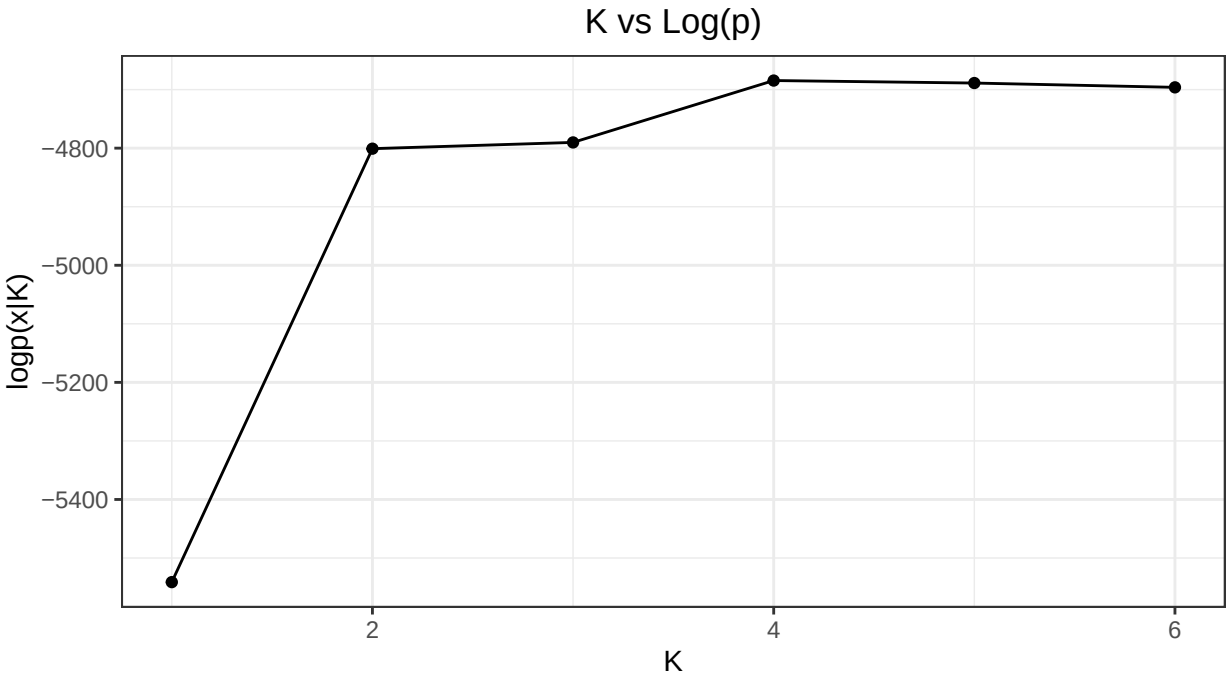
```
ret_6_a = lapply(1:6,gibbs_sampler_n,V=c(2,2),x=data_3_a,round=6000) # sample
K_6_a = sapply(1:6,function(K) logp_x(data_3_a,K,c(2,2),ret_6_a[[K]])) # calculate logp
ggdf_6a = data.frame(cbind(logp = K_6_a, K = 1:6)) # construct df
ggplot(data = ggdf_6a, aes(x = K, y = logp)) +
  geom_line() + geom_point() +
  ggtitle("K vs Log(p)") +
  theme_bw() +
  theme(plot.title = element_text(hjust = 0.5)) +
  xlab("K") + ylab("logp(x|K)") + theme(aspect.ratio=3/6)
```



$\log \tilde{p}\left(x_{\text{obs}} \mid K\right)$ reaches its maximum when K=2 and then starts to decrease. Therefore, we would pick $K = 2$ in this case. The choice is also consistent with our simulation that in total of two subspeicies are defined.

(b)

```
ret_6_b = lapply(1:6,gibbs_sampler_n,V=V,x=data,round=6000) # sample
K_6_b = sapply(1:6,function(K) logp_x(data,K,V,ret_6_b[[K]])) # calculate logp
ggdf_6b = data.frame(cbind(logp = K_6_b, K = 1:6)) # construct df
ggplot(data = ggdf_6b, aes(x = K, y = logp))+
  geom_line() + geom_point()+
  ggtitle("K vs Log(p)") +
  theme_bw() +
  theme(plot.title = element_text(hjust = 0.5)) +
  xlab("K") + ylab("logp(x|K)") + theme(aspect.ratio=3/6)
```



Based on the plot, we can narrow down the choices of K into $K = 2, 3, 4$ for we can observe the first-time level off at $K = 2$ and the second-time level off at $K = 4$, and the value at $K = 4$ is not that significantly larger than when $K = 2$. $K = 2$ can capture the difference between "common impala" and "black-faced impala", but based on our observation in Q4 and Q5,subspecies under the black-faced impala is non-negligible. If we pick $K = 4$, still based on our visualization in Q4, we can only observe 3 main components even if we set $K = 4$ in that question. Therefore, we would still pick $K = 3$ in this case. The plot is slighly out of the expectation for there should be a level off at $K = 3$. This does't show up in that way either because of the inaccurate implementation of the functions or because of the noises and the insignificant differences between the subspecies of black-faced impala.