AML_Lab2

CHAO FU

9/16/2021

1

```
## [1] "The information of HMM model is:"
```

```
## $States
##  [1] "A" "B" "C" "D" "E" "F" "G" "H" "I" "J"
##
## $Symbols
##  [1] "A" "B" "C" "D" "E" "F" "G" "H" "I" "J"
##
## $startProbs
##   A   B   C   D   E   F   G   H   I   J
## 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1
##
## $transProbs
##     to
## from   A   B   C   D   E   F   G   H   I   J
##    A 0.5 0.5 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
##    B 0.0 0.5 0.5 0.0 0.0 0.0 0.0 0.0 0.0 0.0
##    C 0.0 0.0 0.5 0.5 0.0 0.0 0.0 0.0 0.0 0.0
##    D 0.0 0.0 0.0 0.5 0.5 0.0 0.0 0.0 0.0 0.0
##    E 0.0 0.0 0.0 0.0 0.5 0.5 0.0 0.0 0.0 0.0
##    F 0.0 0.0 0.0 0.0 0.0 0.5 0.5 0.0 0.0 0.0
##    G 0.0 0.0 0.0 0.0 0.0 0.0 0.5 0.5 0.0 0.0
##    H 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.5 0.5 0.0
##    I 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.5 0.5
##    J 0.5 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.5
##
## $emissionProbs
##       symbols
## states   A   B   C   D   E   F   G   H   I   J
##      A 0.2 0.2 0.2 0.0 0.0 0.0 0.0 0.0 0.2 0.2
##      B 0.2 0.2 0.2 0.2 0.0 0.0 0.0 0.0 0.0 0.2
##      C 0.2 0.2 0.2 0.2 0.2 0.0 0.0 0.0 0.0 0.0
##      D 0.0 0.2 0.2 0.2 0.2 0.2 0.0 0.0 0.0 0.0
##      E 0.0 0.0 0.2 0.2 0.2 0.2 0.2 0.0 0.0 0.0
##      F 0.0 0.0 0.0 0.2 0.2 0.2 0.2 0.2 0.0 0.0
##      G 0.0 0.0 0.0 0.0 0.2 0.2 0.2 0.2 0.2 0.0
##      H 0.0 0.0 0.0 0.0 0.0 0.2 0.2 0.2 0.2 0.2
##      I 0.2 0.0 0.0 0.0 0.0 0.0 0.2 0.2 0.2 0.2
##      J 0.2 0.2 0.0 0.0 0.0 0.0 0.0 0.2 0.2 0.2
```

2

```
## [1] "The result of simulated HMM model is:"
```

```
## $states
##   [1] "G" "G" "G" "G" "H" "H" "I" "I" "I" "J" "J" "A" "B" "B" "C" "C" "D" "E"
##  [19] "E" "E" "E" "E" "E" "F" "G" "H" "H" "I" "I" "I" "I" "J" "J" "J" "A" "A"
##  [37] "B" "B" "C" "C" "D" "D" "E" "F" "G" "G" "H" "I" "I" "I" "J" "J" "J" "J"
##  [55] "J" "A" "B" "C" "D" "E" "E" "F" "G" "H" "H" "I" "J" "J" "J" "J" "A" "A"
##  [73] "B" "C" "D" "E" "F" "F" "F" "F" "G" "H" "I" "I" "I" "J" "A" "A" "B" "B"
##  [91] "C" "C" "D" "D" "D" "D" "E" "F" "F" "G"
##
## $observation
##   [1] "G" "I" "G" "E" "J" "J" "J" "G" "H" "I" "H" "B" "D" "A" "E" "A" "E" "D"
##  [19] "F" "G" "G" "C" "F" "G" "H" "I" "F" "G" "A" "G" "A" "J" "H" "J" "C" "C"
##  [37] "J" "D" "A" "B" "E" "B" "C" "F" "H" "E" "J" "H" "J" "A" "H" "J" "B" "I"
##  [55] "H" "C" "D" "D" "E" "D" "F" "F" "E" "G" "G" "I" "J" "J" "J" "A" "I" "J"
##  [73] "D" "A" "D" "G" "D" "G" "E" "D" "H" "I" "G" "H" "A" "H" "J" "B" "J" "C"
##  [91] "A" "C" "F" "C" "C" "B" "F" "E" "G" "H"
```

3

```
## [1] "The partial filter probability is:"
```

```
##       index
## states   1         2   3 4 5   6    7    8     9     10
##      A 0.0 0.0000000 0.0 0 0 0.0 0.00 0.00 0.000 0.1875
##      B 0.0 0.0000000 0.0 0 0 0.0 0.00 0.00 0.000 0.0000
##      C 0.0 0.0000000 0.0 0 0 0.0 0.00 0.00 0.000 0.0000
##      D 0.0 0.0000000 0.0 0 0 0.0 0.00 0.00 0.000 0.0000
##      E 0.2 0.0000000 0.0 0 0 0.0 0.00 0.00 0.000 0.0000
##      F 0.2 0.0000000 0.0 0 0 0.0 0.00 0.00 0.000 0.0000
##      G 0.2 0.2857143 0.2 1 0 0.0 0.00 0.00 0.000 0.0000
##      H 0.2 0.2857143 0.4 0 1 0.5 0.25 0.25 0.125 0.0625
##      I 0.2 0.2857143 0.4 0 0 0.5 0.50 0.75 0.500 0.3125
##      J 0.0 0.1428571 0.0 0 0 0.0 0.25 0.00 0.375 0.4375
```

```
## [1] "The partial smooth probability is:"
```

```
##       index
## states   1 2 3 4 5    6   7    8          9         10
##      A 0.0 0 0 0 0 0.00 0.0 0.00 0.00000000 0.0000000
##      B 0.0 0 0 0 0 0.00 0.0 0.00 0.00000000 0.0000000
##      C 0.0 0 0 0 0 0.00 0.0 0.00 0.00000000 0.0000000
##      D 0.0 0 0 0 0 0.00 0.0 0.00 0.00000000 0.0000000
##      E 0.0 0 0 0 0 0.00 0.0 0.00 0.00000000 0.0000000
##      F 0.5 0 0 0 0 0.00 0.0 0.00 0.00000000 0.0000000
##      G 0.5 1 1 1 0 0.00 0.0 0.00 0.00000000 0.0000000
##      H 0.0 0 0 0 1 0.75 0.5 0.25 0.08333333 0.0000000
##      I 0.0 0 0 0 0 0.25 0.5 0.75 0.66666667 0.4166667
##      J 0.0 0 0 0 0 0.00 0.0 0.00 0.25000000 0.5833333
```

```
## [1] "The probable path is:"
```

```
##   [1] "F" "G" "G" "G" "H" "H" "H" "H" "H" "I" "J" "A" "B" "B" "C" "C" "C" "C"
##  [19] "D" "E" "E" "E" "E" "E" "F" "G" "G" "H" "I" "I" "I" "I" "J" "A" "A" "A"
##  [37] "A" "B" "B" "B" "C" "C" "D" "E" "F" "G" "H" "H" "H" "I" "I" "I" "J" "J"
##  [55] "J" "A" "B" "B" "C" "D" "E" "F" "G" "H" "I" "J" "A" "A" "A" "A" "A" "A"
```

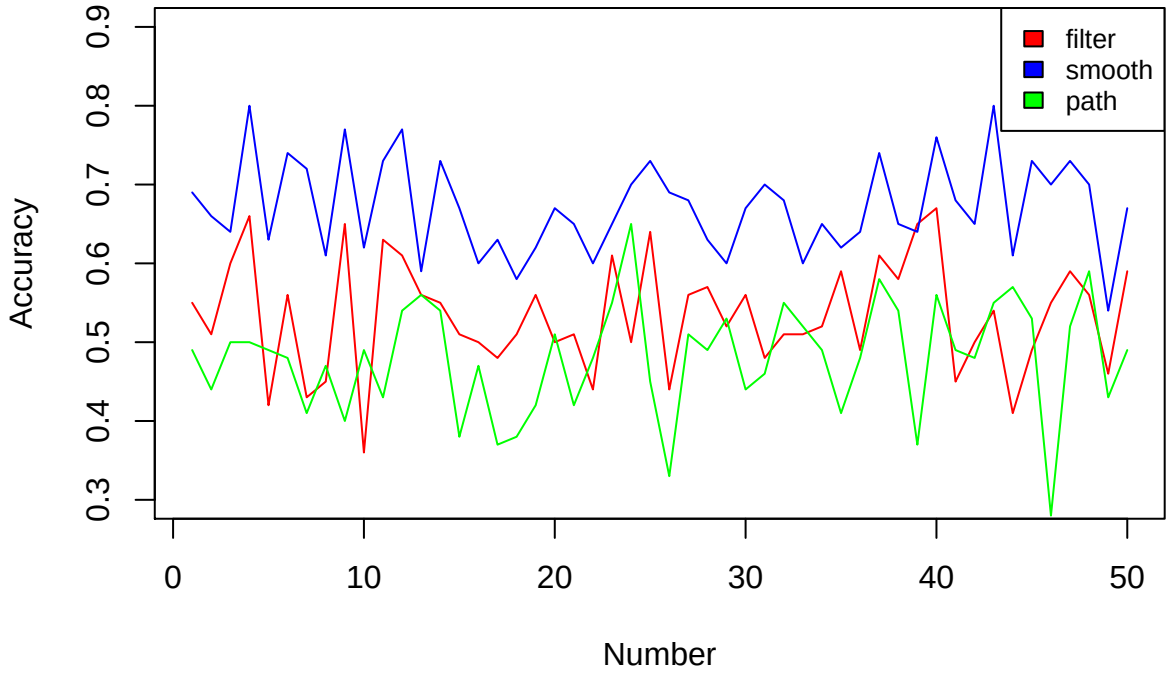
```
##  [73] "B" "C" "D" "E" "E" "E" "E" "E" "F" "G" "G" "H" "I" "J" "A" "A" "A" "A"
##  [91] "B" "C" "D" "D" "D" "D" "D" "D" "E" "F"
```

4

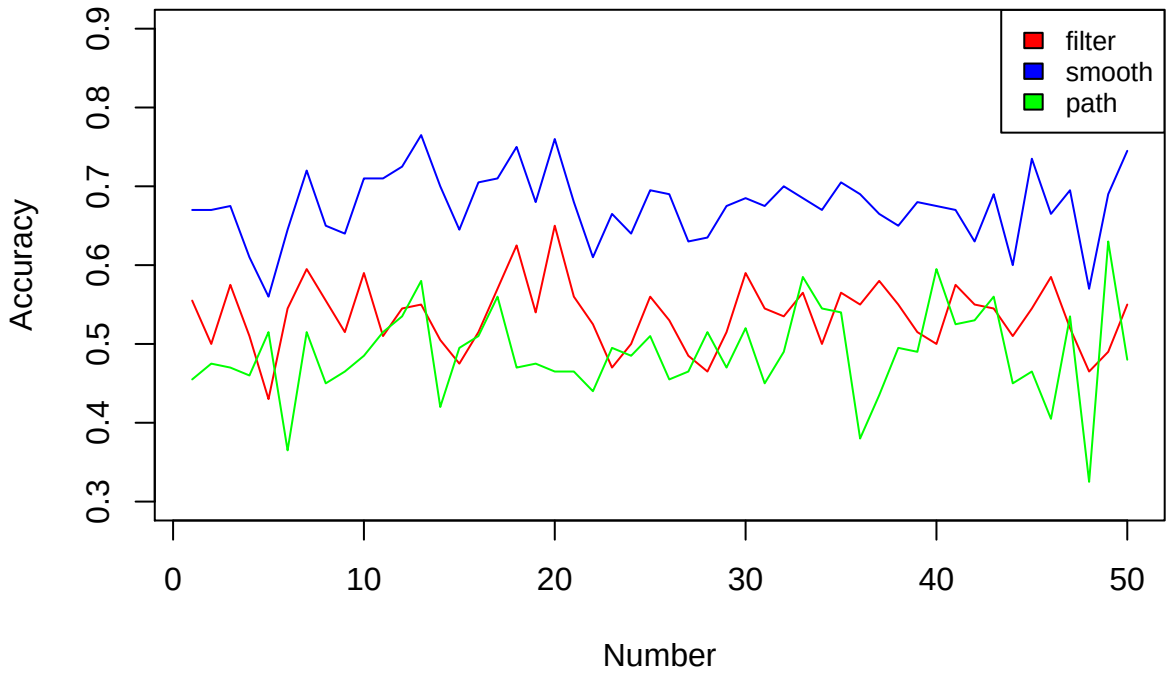
```
## [1] "The filter accuracy is:"
## [1] 0.55
## [1] "The smooth accuracy is:"
## [1] 0.7
## [1] "The probable path accuracy is:"
## [1] 0.49
```

5

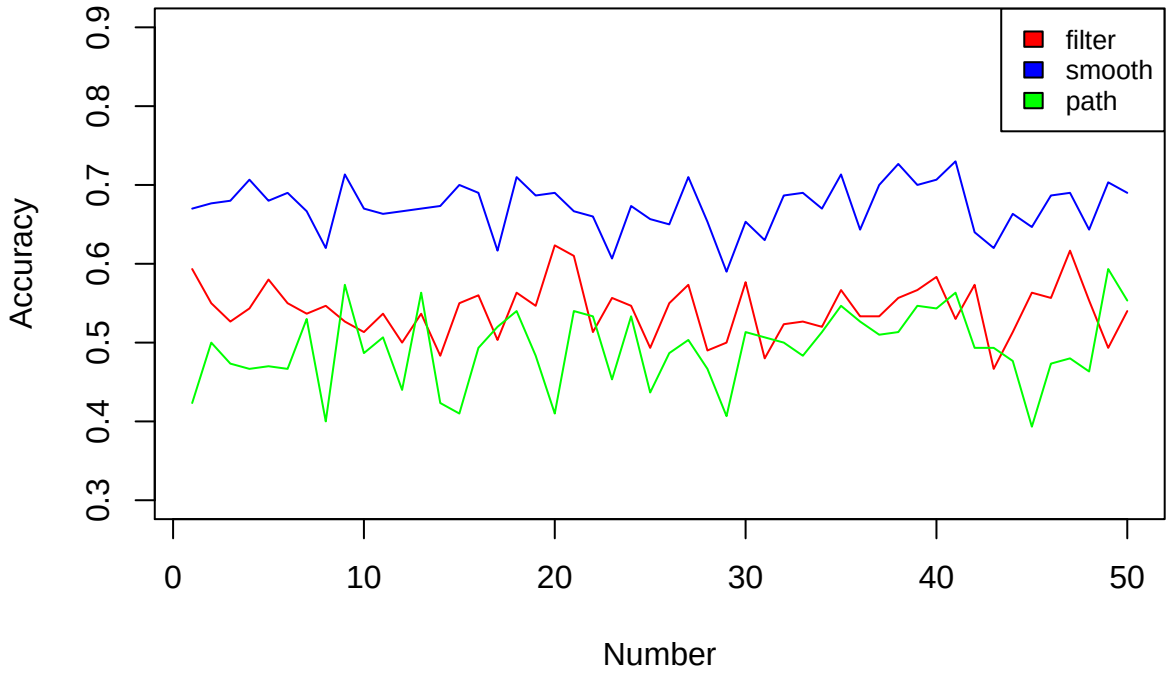
```
## [1] "The comprasion accuracy with 50 times and 100 samples"
```



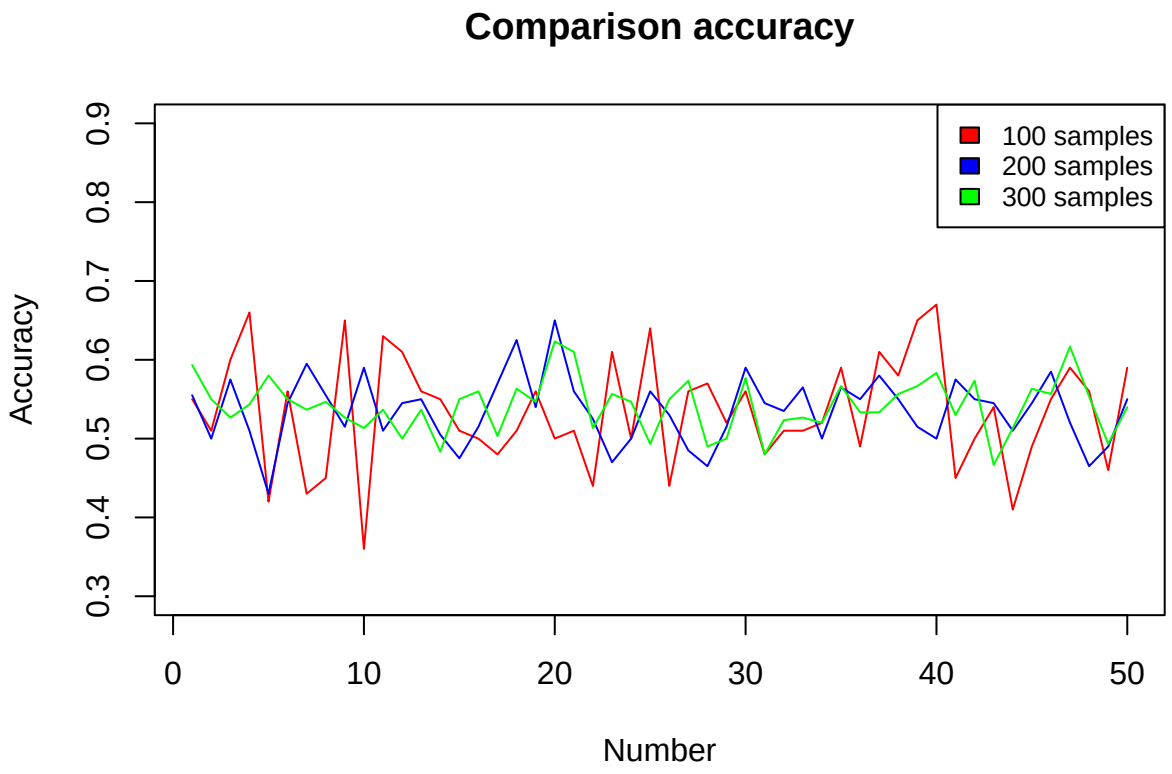

```
## [1] "The comprasion accuracy with 50 times and 200 samples"
```



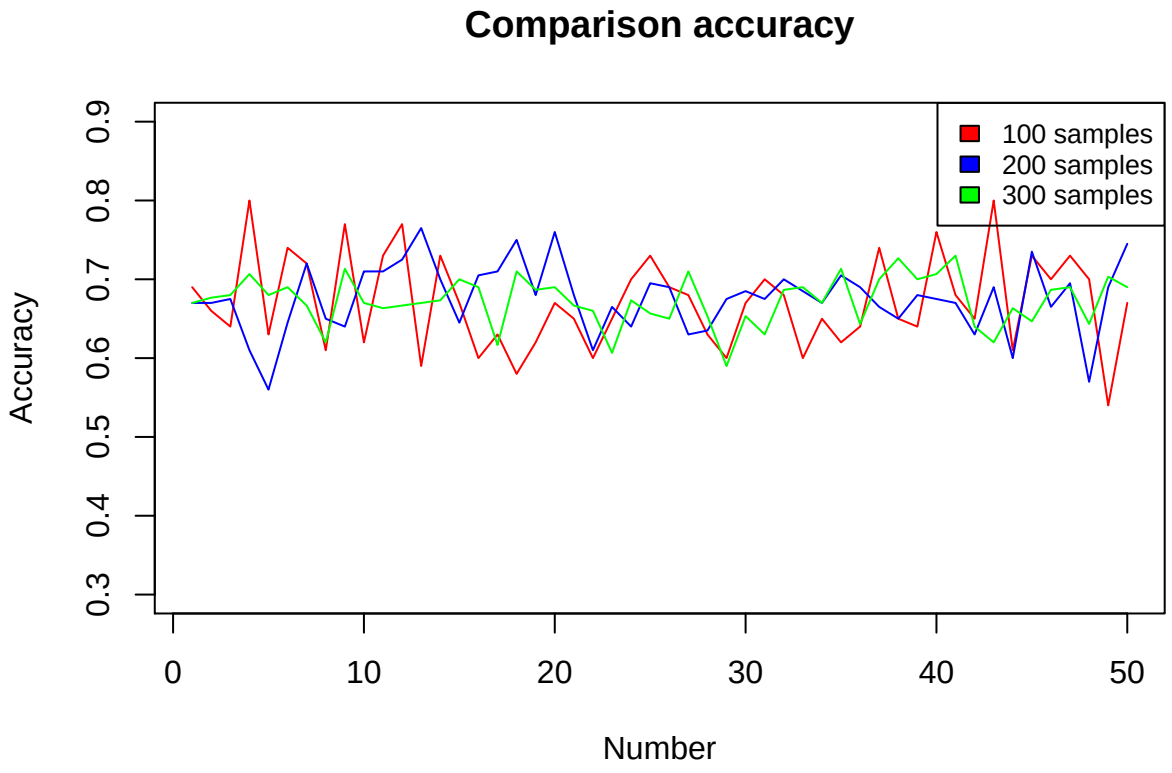

```
## [1] "The comprasion accuracy with 50 times and 300 samples"
```



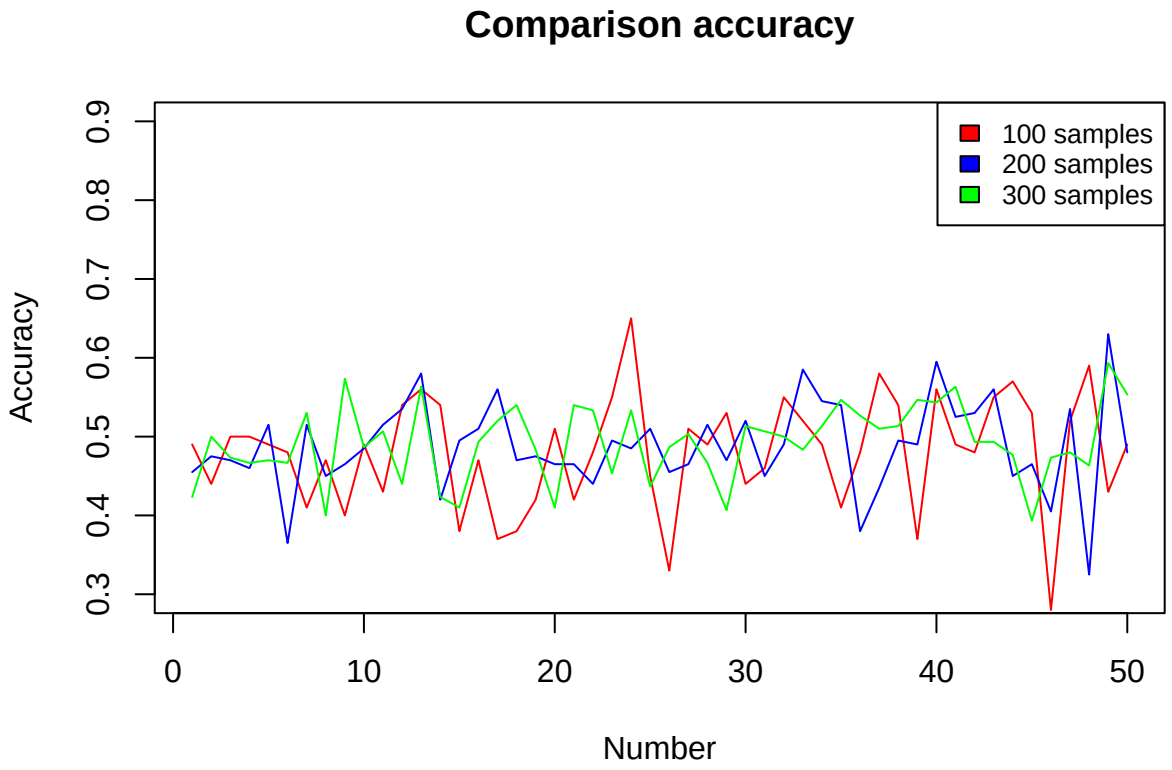

```
## [1] "The comprasion filter accuracy with different sample size"
```




```
## [1] "The comprasion smooth accuracy with different sample size"
```




```
## [1] "The comprasion path accuracy with different sample size"
```



Comment

1)

The filtered distribution notation is given below:

$$p(z^t|x^{0:t})$$

Filtering is the posterior probability of hidden states given by observations in different time steps. More observations can get greater estimation of posterior probability. However, the number of observations which the hidden states can use is increasing with time steps. The hidden states in the former time steps can use fewer observations than the latter ones. Hence, the latter time steps can have better estimations of posterior probability than the former ones.

The smoothing distribution notation is given below:

$$p(z^t|x^{0:T})$$

Smoothing is also the posterior probability of hidden states given by observations. The difference is that smoothing use all the observations to calculate posterior probability in each time step. Hence, smoothing have better estimations and accuracy than filtering.

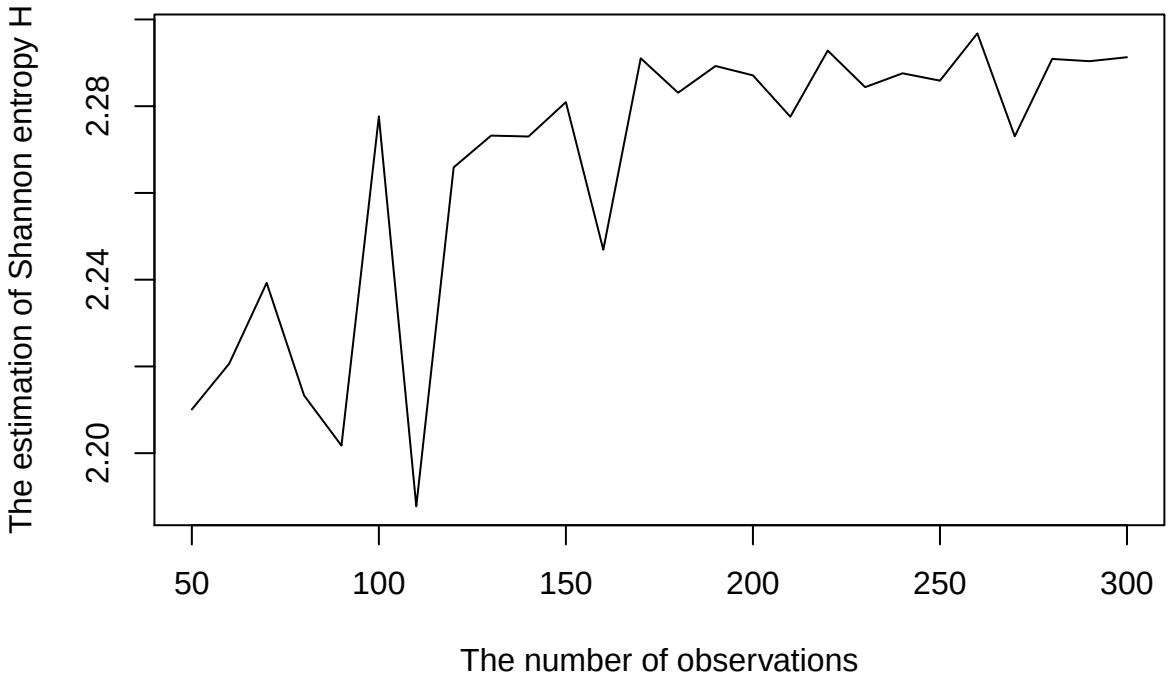
2)

The posterior probability of hidden states given by observations can be used to find the most probable path from the first time step to the end. The Viterbi algorithm(partial formula) is given below:

$$w(z^{t+1}) := logp(x^{t+1}|z^{t+1}) + max_{z^t}[logp(z^{t+1}|z^t) + w(t)(t = 0, 1, \ldots, T-1)$$

It can be seen that the most probable hidden state in each time step is selected by previous observations not all. Hence, the accuracy of this method is lower than smoothing.

6



Comment

The Shannon entropy H has a significant fluctuation with the increasing of observations. The large Shannon entropy value means less information. Hence, it is difficult to know the location of the robot. In the graph, it can be seen that the Shannon entropy has a growing trend after 110 observations. Hence, the more observations can not be better to know where the robot is.

7

```
## [1] "The probabilites of the hidden states for the time step 101 is:"
```

```
##          A          B          C          D          E          F          G
## 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.27272727 0.45454545
##          H          I          J
## 0.22727273 0.04545455 0.00000000
```

Appendix

```
knitr::opts_chunk$set(echo = TRUE)
library(HMM)
library(entropy)
#set stats and symbols alphabet
unob <- LETTERS[1:10]
ob <- LETTERS[1:10]
#set start probability
startprob <- rep(0.1, 10)
#set transition probability
prob1 <- matrix(0, 10, 10)
for (i in 1:10){
  if (i <= 9){
    prob1[i, c(i, i + 1)] <- c(0.5, 0.5)
```

```
  }else {prob1[i, c(i, i - 9)] <- c(0.5, 0.5)}
}
#set emission probability
prob2 <- matrix(0, 10, 10)
for (i in 3:8){
  prob2[i, (i - 2) : (i + 2)] <- rep(0.2, 5)
}
c1 <- rep(c(0.2, 0, 0.2), c(3, 5, 2))
c2 <- rep(c(0.2, 0, 0.2), c(4, 5, 1))
prob2[1, ] <- c1
prob2[2, ] <- c2
prob2[9, ] <- rev(c2)
prob2[10, ] <- rev(c1)
#set initial HMM
my_hmm <- initHMM(States = unob, Symbols = ob, startProbs = startprob,
                  transProbs = prob1, emissionProbs = prob2)
#present initial HMM
print("The information of HMM model is:")
cat("\n")
my_hmm
#set random seed
set.seed(4)
#simulate HMM for 100 time steps
simu_hmm <- simHMM(my_hmm, 100)
#present path for states and symbols
print("The result of simulated HMM model is:")
cat("\n")
simu_hmm
#extract observation sample
my_ob_sample <- simu_hmm$observation
#calculate filter probability (alpha)
my_forward <- forward(my_hmm, my_ob_sample)
my_filter <- prop.table(exp(my_forward), 2)
#present first 10 filter probability
print("The partial filter probability is:")
cat("\n")
print(my_filter[, 1:10])
cat("\n")
#calculate backward probability (beta)
my_backward <- backward(my_hmm, my_ob_sample)
#calculate forward-backward probability (alpha * beta)
my_for_back <- exp(my_forward) * exp(my_backward)
#calculate smooth probability
my_smooth <- prop.table(my_for_back, 2)
#present first 10 smooth probability
print("The partial smooth probability is:")
cat("\n")
print(my_smooth[, 1:10])
cat("\n")
#use posterior function to calculate smooth
#posterior(my_hmm, my_ob_sample)
#calculate probable path of states by viterbi algorithm
my_path <- viterbi(my_hmm, my_ob_sample)
```

```r
#present probable path of states
print("The probable path is:")
cat("\n")
print(my_path)
#extract states sample
my_unob_sample <- simu_hmm$states
#set function to extract most probability states
my_most_states <- function(my_post, marginal){
  return(unname(apply(my_post, marginal, function(x) names(which.max(x)))))
}
#calculate filter and smooth most probability states
my_filter_post <- my_most_states(my_filter, 2)
my_smooth_post <- my_most_states(my_smooth, 2)
#set function to calculate accuracy between true hidden states and posterior
my_accuracy <- function(state1, state2){
  my_table <- table(state1, state2)
  accu <- sum(diag(my_table))/ sum(my_table)
  return(accu)
}
#calculate filter accuracy
print("The filter accuracy is:")
cat("\n")
print(my_accuracy(my_unob_sample, my_filter_post))
cat("\n")
#calculate smooth accuracy
print("The smooth accuracy is:")
cat("\n")
print(my_accuracy(my_unob_sample, my_smooth_post))
cat("\n")
#calculate most probable path
print("The probable path accuracy is:")
cat("\n")
print(my_accuracy(my_unob_sample, my_path))
#set random seed
set.seed(4)
#set function to calculate accuracy with different simulated samples
multi_accuray <- function(iter, size){
  #generate samples
  sample_all <- lapply(seq_len(iter), function(x) simHMM(my_hmm, size))
  sample_list <- lapply(sample_all, function(x) list("sta" = x$states, "obs" = x$observation))
  #calculate most probability states in filter
  sample_filter <- lapply(sample_list, function(x) my_most_states(prop.table(exp(forward(my_hmm, x$obs)
  #calculate most probability states in smooth
  sample_smooth <- lapply(sample_list, function(x) my_most_states(posterior(my_hmm, x$obs), 2))
  #calculate most probable path
  sample_path <- lapply(sample_list, function(x) viterbi(my_hmm, x$obs))
  #calculate filter accuracy
  accuracy_filter <- mapply(function(x, y) my_accuracy(x$sta, y), sample_list, sample_filter)
  #calculate smooth accuracy
  accuracy_smooth <- mapply(function(x, y) my_accuracy(x$sta, y), sample_list, sample_smooth)
  #calculate path accuracy
  accuracy_path <- mapply(function(x, y) my_accuracy(x$sta, y), sample_list, sample_path)
  #output
```

```
  return(data.frame("filter" = accuracy_filter, "smooth" = accuracy_smooth, "path" = accuracy_path))
}
#calculate accuracy with different size of simulated HMM
mul100 <- multi_accuray(iter = 50, size = 100)
mul200 <- multi_accuray(iter = 50, size = 200)
mul300 <- multi_accuray(iter = 50, size = 300)
#set plot function
my_plot <- function(n, d){
  plot(x = seq_len(n), y = d$filter, type = "l", col = "red",
       xlab = "Number", ylab = "Accuracy", ylim = c(0.3, 0.9))
  lines(x = seq_len(n), y = d$smooth, type = "l", col = "blue")
  lines(x = seq_len(n), y = d$path, type = "l", col = "green")
  legend("topright", legend = c("filter", "smooth", "path"),
         fill = c("red", "blue","green"), cex = 0.8)
}
#set plot function
my_plot1 <- function(n, d1, d2, d3){
  plot(x = seq_len(n), y = d1, type = "l", col = "red",
       xlab = "Number", ylab = "Accuracy", main = "Comparison accuracy",
       ylim = c(0.3, 0.9))
  lines(x = seq_len(n), y = d2, type = "l", col = "blue")
  lines(x = seq_len(n), y = d3, type = "l", col = "green")
  legend("topright", legend = c("100 samples", "200 samples", "300 samples"),
         fill = c("red", "blue","green"), cex = 0.8)
}
print("The comprasion accuracy with 50 times and 100 samples")
my_plot(50, mul100)
cat("\n")
print("The comprasion accuracy with 50 times and 200 samples")
my_plot(50, mul200)
cat("\n")
print("The comprasion accuracy with 50 times and 300 samples")
my_plot(50, mul300)
cat("\n")
print("The comprasion filter accuracy with different sample size")
my_plot1(50, mul100$filter, mul200$filter, mul300$filter)
cat("\n")
print("The comprasion smooth accuracy with different sample size")
my_plot1(50, mul100$smooth, mul200$smooth, mul300$smooth)
cat("\n")
print("The comprasion path accuracy with different sample size")
my_plot1(50, mul100$path, mul200$path, mul300$path)
set.seed(4)
entro_sample <- lapply(seq(50, 300, 10), function(x) table(simHMM(my_hmm, x)$observation))
entro_result <- lapply(entro_sample, function(x) entropy.empirical(x))
plot(x = seq(50, 300, 10), unlist(entro_result), type = "l",
     xlab = "The number of observations",
     ylab = "The estimation of Shannon entropy H")

#get the 100th filter probability
pre_filter <- my_filter[, 100]
#get transition matrix
transi_matrix <- prob1
```

```r
#calculate the prediction probabilities
pre_prob <- apply(transi_matrix, 2, function(x) sum(x * pre_filter))
names(pre_prob) <- names(pre_filter)
#present the result
print("The probabilites of the hidden states for the time step 101 is:")
cat("\n")
print(pre_prob)
```