AML Lab 2

Harshavardhan Subramanian

22/09/2020

1.

```
## The built Hidden Markov Model is:

## $States
##  [1] "A" "B" "C" "D" "E" "F" "G" "H" "I" "J"
##
## $Symbols
##  [1] "x1"  "x2"  "x3"  "x4"  "x5"  "x6"  "x7"  "x8"  "x9"  "x10"
##
## $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  x1  x2  x3  x4  x5  x6  x7  x8  x9 x10
##      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
```

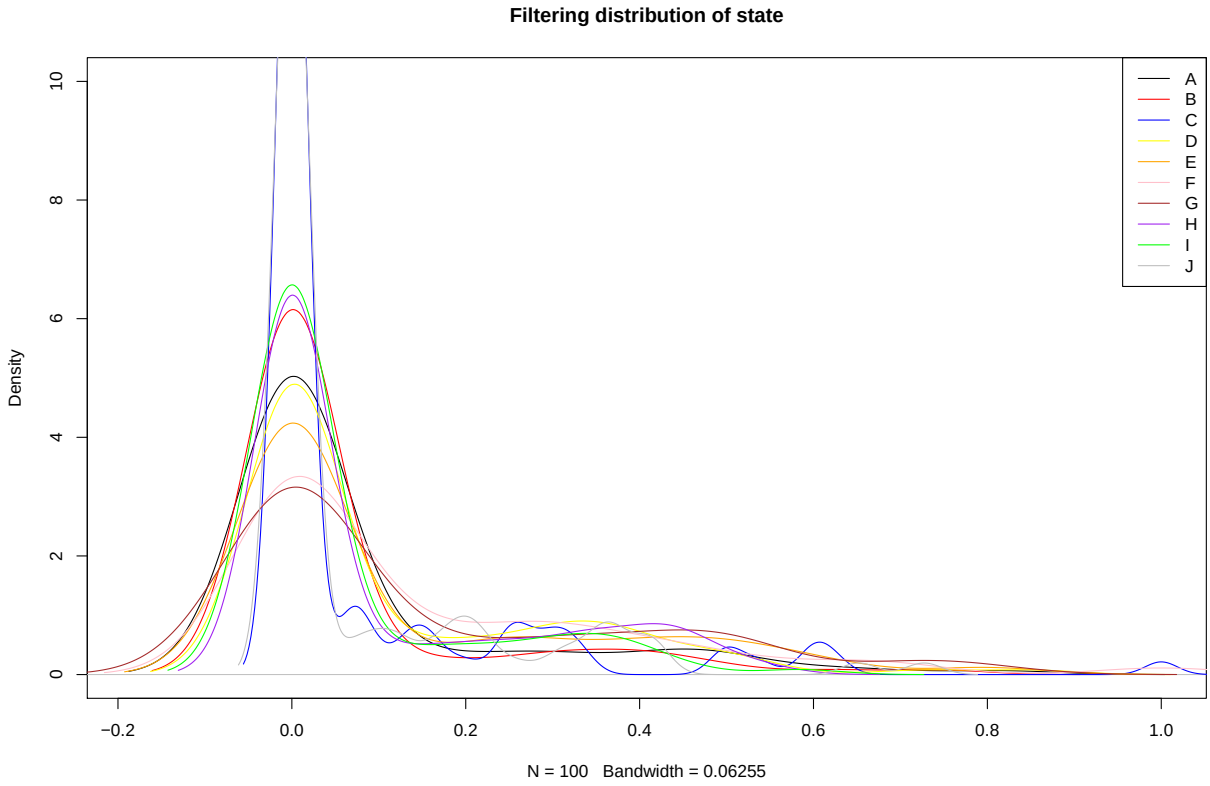
2

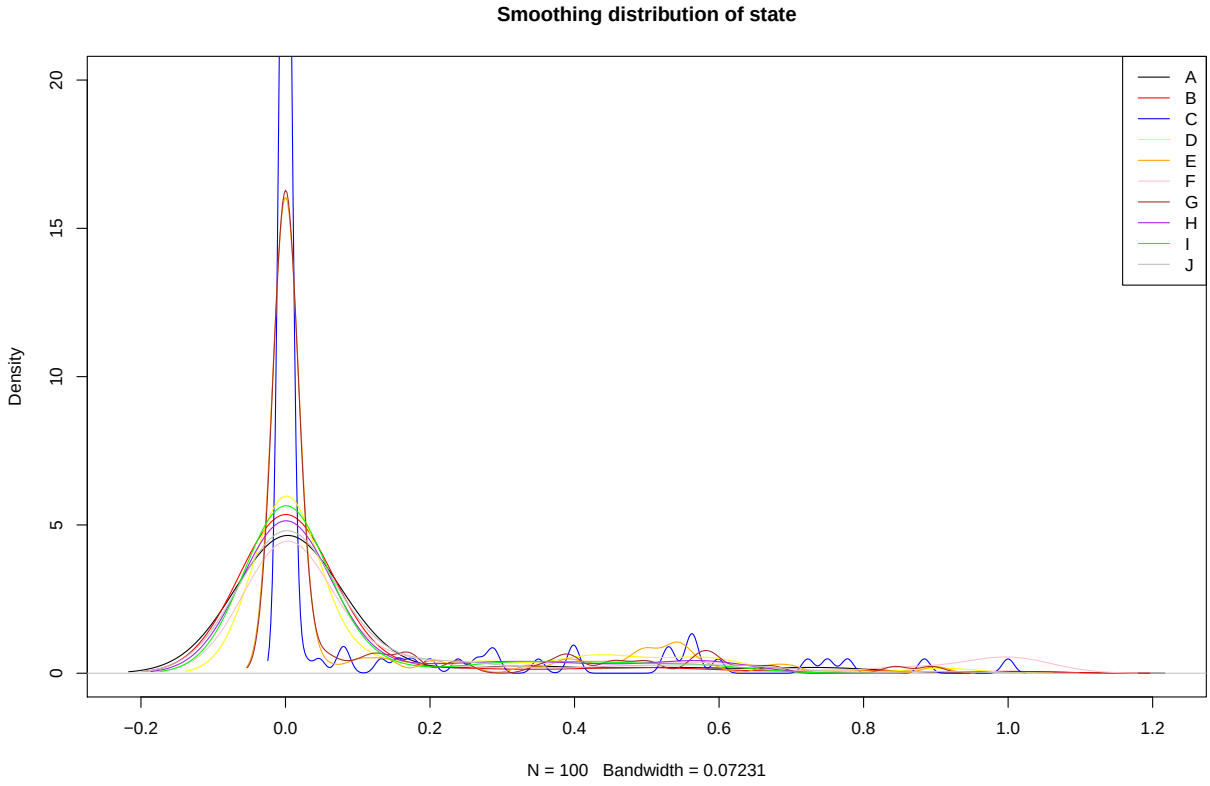
```
##
##  The Output of Simulated HMM model for 100 Timesteps:

## $states
```

```
##   [1] "E" "E" "F" "G" "G" "G" "H" "I" "I" "J" "J" "J" "J" "J" "J" "A" "B"
##  [18] "C" "D" "D" "E" "F" "G" "G" "H" "I" "J" "A" "B" "C" "D" "D" "E" "E"
##  [35] "E" "E" "F" "F" "F" "G" "G" "H" "I" "J" "A" "A" "B" "C" "C" "D" "E"
##  [52] "E" "F" "F" "G" "G" "H" "I" "J" "A" "A" "B" "C" "C" "D" "D" "E" "E"
##  [69] "F" "G" "G" "H" "H" "I" "J" "A" "B" "C" "C" "D" "E" "E" "F" "F" "G"
##  [86] "H" "I" "J" "A" "A" "A" "A" "A" "A" "B" "C" "D" "D" "D" "E"
##
## $observation
##   [1] "x3"  "x3"  "x4"  "x6"  "x8"  "x7"  "x9"  "x1"  "x9"  "x9"  "x9"
##  [12] "x8"  "x1"  "x10" "x1"  "x1"  "x3"  "x2"  "x3"  "x2"  "x6"  "x8"
##  [23] "x8"  "x8"  "x6"  "x9"  "x10" "x1"  "x1"  "x3"  "x5"  "x5"  "x5"
##  [34] "x7"  "x6"  "x5"  "x8"  "x4"  "x8"  "x9"  "x5"  "x8"  "x9"  "x2"
##  [45] "x1"  "x2"  "x3"  "x2"  "x4"  "x4"  "x6"  "x6"  "x6"  "x4"  "x8"
##  [56] "x8"  "x8"  "x9"  "x8"  "x9"  "x2"  "x10" "x5"  "x1"  "x4"  "x5"
##  [67] "x3"  "x5"  "x6"  "x7"  "x9"  "x7"  "x7"  "x7"  "x1"  "x9"  "x3"
##  [78] "x2"  "x3"  "x5"  "x7"  "x6"  "x4"  "x4"  "x8"  "x10" "x8"  "x2"
##  [89] "x10" "x1"  "x9"  "x10" "x9"  "x10" "x4"  "x3"  "x4"  "x4"  "x2"
## [100] "x3"
```

3 & 4


Filtering distribution of state
Density
0
2
4
6
8
10
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
A
B
C
D
E
F
G
H
I
J
N = 100 Bandwidth = 0.06255


Smoothing distribution of state
Density
0
5
10
15
20
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
1.2
A
B
C
D
E
F
G
H
I
J
N = 100 Bandwidth = 0.07231

```
##
##  The most probable path is :
## A B C D E F F G H I J A A A A A A B C C C C C C D D D E F F F F F F F F F F G H I J A A A A A A B C C
##
## The accuracy of filtered distribution is found to be :
## [1] 0.47
##
##  The accuracy of smoother distribution is found to be :
## [1] 0.66
##
##  The accuracy of most probable path is found to be :
## [1] 0.44
```

5

```
##
##  Accuracies obtained when the number of time steps is 25
## $filter_acc
## [1] 0.36
##
## $smoother_acc
## [1] 0.76
##
## $most_prob_acc
## [1] 0.6
##
##  Accuracies obtained when the number of time steps is 70
## $filter_acc
## [1] 0.6142857
##
## $smoother_acc
## [1] 0.8142857
##
## $most_prob_acc
## [1] 0.5428571
##
##  Accuracies obtained when the number of time steps is 120
## $filter_acc
## [1] 0.5166667
##
## $smoother_acc
## [1] 0.6833333
##
## $most_prob_acc
## [1] 0.6166667
##
##  Accuracies obtained when the number of time steps is 200
```

```
## $filter_acc
## [1] 0.495
##
## $smoother_acc
## [1] 0.73
##
## $most_prob_acc
## [1] 0.475
```

It is evident from the above results that the accuracy of Smoother distributions is higher than both Filtered distribution and the most probable path as well by considering different number of Time steps.

This could be because in Smoother distribution, the probability of hidden variables is calculated for all the Time steps considering the number of observations is higher than the number of time steps. This gives an edge over filtered distribution in which the the current probability of hidden state depends on all the previous point until that point. Meaning the number of Time steps is equal to number of Observations.

Where as in Most Probable path found by Viterbi algorithm,it takes into account that the probabilty of robot being in that state is maximized and it is the maximum probable state in each step at the end. This could be an hindrance in finding most suitable state collectively as it finds most probable state in each time step.

6

```
##   150 Timesteps 200 Timesteps 300 Timesteps 350 Timesteps
## 1      2.288059      2.286014       2.29687      2.293897
```

From the above table it is evident that the entropy for different time steps is pretty much the same. This suggests that the position of robot is better or not, cannot be identified by increasing time steps. This could be because, the parameters are the same irrespective of simulating for different number of time steps.

7

```
## The probabilities of the hidden states for 101 time step are:
```

```
##      [,1]       [,2]      [,3]      [,4]      [,5]      [,6] [,7] [,8]
## [1,]    0 0.01883562 0.1215753 0.3339041 0.3784247 0.1472603    0    0
##      [,9] [,10]
## [1,]    0     0
```

Appendix

```
knitr::opts_chunk$set(echo = TRUE,warning = FALSE, message = FALSE,fig.width=12, fig.height=8)
library(HMM)
trans_prob <- matrix(0,10,10)
for(i in seq(1,nrow(trans_prob))){
  trans_prob[i,i] <- 0.5
  if(i == 10)
  {
    trans_prob[i,1] <- 0.5
  }
  else
  {
    trans_prob[i,i+1] <- 0.5
  }
}
```

```
emiss_prob <- matrix(0,10,10)
for(i in seq(1,nrow(emiss_prob))){

  if(i == 1)
  {
    emiss_prob[i,i] <- 0.2
    emiss_prob[i,nrow(emiss_prob)] <- 0.2
    emiss_prob[i,nrow(emiss_prob)-1] <- 0.2
    emiss_prob[i,i+1] <- 0.2
    emiss_prob[i,i+2] <- 0.2

  }
  else if(i == 2)
  {
    emiss_prob[i,i] <- 0.2
    emiss_prob[i,i-1] <- 0.2
    emiss_prob[i,nrow(emiss_prob)] <- 0.2
    emiss_prob[i,i+1] <- 0.2
    emiss_prob[i,i+2] <- 0.2

  }

  else if(i >= 3 && i <= 8)
  {
    emiss_prob[i,i+1] <- 0.2
    emiss_prob[i,i+2] <- 0.2
    emiss_prob[i,i] <- 0.2
    emiss_prob[i,i-1] <- 0.2
    emiss_prob[i,i-2] <- 0.2
  }
  else if(i == 9)
  {
    emiss_prob[i,i] <- 0.2
    emiss_prob[i,i-1] <- 0.2
    emiss_prob[i,i-2] <- 0.2
    emiss_prob[i,i + 1] <- 0.2
    emiss_prob[i,1] <- 0.2

  }
  else
  {
    emiss_prob[i,i] <- 0.2
    emiss_prob[i,i-1] <- 0.2
    emiss_prob[i,i-2] <- 0.2
    emiss_prob[i,1] <- 0.2
    emiss_prob[i,2] <- 0.2
  }
}


HMM_mod <- initHMM(c("A","B","C","D","E","F","G","H","I","J"),
        c("x1","x2","x3","x4","x5","x6","x7","x8","x9","x10"),
        c(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.10,0.1),
```

```
        trans_prob,
        emiss_prob)
cat("The built Hidden Markov Model is:")
HMM_mod

HMM_models <- function(time_steps)
{

sim_HMM <- simHMM(HMM_mod,time_steps)


# 3
alpha <- forward(HMM_mod,sim_HMM$observation)
for_prob <- exp(alpha)

beta <- backward(HMM_mod,sim_HMM$observatio)
back_prob <- exp(beta)

filtering <- matrix(0,10,time_steps)
smoothing <- matrix(0,10,time_steps)
for(j in 1:time_steps)
{
  filtering[,j] <- for_prob[,j] / sum(for_prob[,j])
  smoothing[,j] <- for_prob[,j] * back_prob[,j] / sum(for_prob[,j] * back_prob[,j])
}
most_prob_path <- viterbi(HMM_mod, sim_HMM$observation)
if(time_steps == 100)
{
  colors <- c("Black","Red","Blue","Yellow","Orange","Pink","Brown","Purple","Green","Grey")
  plot(density(filtering[1,]), main = "Filtering distribution of state", ylim = c(0,10))
  for(i in 2:10)
  {
    lines(density(filtering[i,]),col = colors[i])
  }
  legend("topright",legend=paste(c("A","B","C","D","E","F","G","H","I","J")),col=c("Black","Red","Blue"

  plot(density(smoothing[1,]), main = "Smoothing distribution of state", ylim = c(0,20))
  for(i in 2:10)
  {
    lines(density(smoothing[i,]),col = colors[i])
  }
  legend("topright",legend=paste(c("A","B","C","D","E","F","G","H","I","J")),col=c("Black","Red","Blue"

  cat("\n The most probable path is : \n")
  cat(most_prob_path)
  cat("\n \n")



}

# 4
states <- c("A","B","C","D","E","F","G","H","I","J")
```

```
filtering[is.nan(filtering)] <- 0
filter_path <- apply(filtering,2,which.max)
filter_states <- states[filter_path]
cm_filter <- table(filter_states,sim_HMM$states)
filter_acc <- sum(diag(cm_filter)) / sum(cm_filter)

smoothing[is.nan(smoothing)] <- 0
smoother_path <- apply(smoothing,2,which.max)
smoother_states <- states[smoother_path]
cm_smoother <- table(smoother_states,sim_HMM$states)
smoother_acc <- sum(diag(cm_smoother)) / sum(cm_smoother)

cm_most_prob <- table(most_prob_path,sim_HMM$states)
most_prob_acc <- sum(diag(cm_most_prob)) / sum(cm_most_prob)

return(list("filter_acc" = filter_acc, "smoother_acc" = smoother_acc,
"most_prob_acc" = most_prob_acc, "filter_states" = filter_states, "smoother_states" = smoother_states,
"most_prob_states" = most_prob_path))

}

sim_HMM_100 <- simHMM(HMM_mod,100)
cat("\n The Output of Simulated HMM model for 100 Timesteps:")
sim_HMM_100

Prob_dist <- HMM_models(time_steps = 100)
cat("The accuracy of filtered distribution is found to be :")
Prob_dist$filter_acc

cat("\n The accuracy of smoother distribution is found to be :")
Prob_dist$smoother_acc

cat("\n The accuracy of most probable path is found to be :")
Prob_dist$most_prob_acc

cat("\n Accuracies obtained when the number of time steps is 25")
HMM_models(time_steps = 25)[1:3]

cat("\n Accuracies obtained when the number of time steps is 70")
HMM_models(time_steps = 70)[1:3]

cat("\n Accuracies obtained when the number of time steps is 120")
HMM_models(time_steps = 120)[1:3]

cat("\n Accuracies obtained when the number of time steps is 200")
HMM_models(time_steps = 200)[1:3]

library(entropy)

sim_HMM_150 <- HMM_models(time_steps = 150)[4]
sim_HMM_200 <- HMM_models(time_steps = 200)[4]
sim_HMM_300 <- HMM_models(time_steps = 300)[4]
sim_HMM_350 <- HMM_models(time_steps = 350)[4]
```

```r
states <- c("A","B","C","D","E","F","G","H","I","J")
filt_freq_150 <- numeric()
filt_freq_200 <- numeric()
filt_freq_300 <- numeric()
filt_freq_350 <- numeric()
for(i in 1:length(states))
{
  filt_freq_150[i] <- length(which(sim_HMM_150$filter_states == states[i])) /
    length(sim_HMM_150$filter_states)
  filt_freq_200[i] <- length(which(sim_HMM_200$filter_states == states[i])) /
    length(sim_HMM_200$filter_states)
  filt_freq_300[i] <- length(which(sim_HMM_300$filter_states == states[i])) /
    length(sim_HMM_300$filter_states)
  filt_freq_350[i] <- length(which(sim_HMM_350$filter_states == states[i])) /
    length(sim_HMM_350$filter_states)
}

entropy_df <- data.frame(entropy.empirical(filt_freq_150)
                         ,entropy.empirical(filt_freq_200),entropy.empirical(filt_freq_300),
                         entropy.empirical(filt_freq_350))
colnames(entropy_df) <- c("150 Timesteps","200 Timesteps","300 Timesteps","350 Timesteps")
entropy_df

alpha <- forward(HMM_mod,sim_HMM_100$observation)
for_prob <- exp(alpha)

beta <- backward(HMM_mod,sim_HMM_100$observatio)
back_prob <- exp(beta)

filtering <- matrix(0,10,100)
smoothing <- matrix(0,10,100)
for(j in 1:100)
{
  filtering[,j] <- for_prob[,j] / sum(for_prob[,j])
  smoothing[,j] <- for_prob[,j] * back_prob[,j] / sum(for_prob[,j] * back_prob[,j])
}
filter_path <- apply(filtering,2,which.max)
filter_states <- states[filter_path]

Z100 <- filtering[,100]
Prob_Z101 <- Z100 %*% trans_prob
cat("The probabilities of the hidden states for 101 time step are: ")
Prob_Z101
#install.packages("kable")
```