sol_lab2

Isak Berntsson

9/15/2022

Question 1

Creating the probability matricies

```
n = 10



startP = rep(1/n,n)
startP
```

```
##  [1] 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1
```

```
P = diag(n) * .5
# 1,2 2,3 3,4... n-1,n
for( i in 1:(n-1)){
  P[i,i+1]=.5
}
P[n,1] = .5 #can loop around


head(P)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
## [1,]  0.5  0.5  0.0  0.0  0.0  0.0  0.0    0    0     0
## [2,]  0.0  0.5  0.5  0.0  0.0  0.0  0.0    0    0     0
## [3,]  0.0  0.0  0.5  0.5  0.0  0.0  0.0    0    0     0
## [4,]  0.0  0.0  0.0  0.5  0.5  0.0  0.0    0    0     0
## [5,]  0.0  0.0  0.0  0.0  0.5  0.5  0.0    0    0     0
## [6,]  0.0  0.0  0.0  0.0  0.0  0.5  0.5    0    0     0
```

```
emissionP = matrix(data = 0, nrow=n,ncol=n)

for (i in 1:n){
  inds = ((i-2):(i+2))-1
  for ( j in ( inds %% 10 ) ){
    emissionP[i,j+1]=0.2
  }
```

```
}

head(emissionP)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
## [1,]  0.2  0.2  0.2  0.0  0.0  0.0  0.0  0.0  0.2   0.2
## [2,]  0.2  0.2  0.2  0.2  0.0  0.0  0.0  0.0  0.0   0.2
## [3,]  0.2  0.2  0.2  0.2  0.2  0.0  0.0  0.0  0.0   0.0
## [4,]  0.0  0.2  0.2  0.2  0.2  0.2  0.0  0.0  0.0   0.0
## [5,]  0.0  0.0  0.2  0.2  0.2  0.2  0.2  0.0  0.0   0.0
## [6,]  0.0  0.0  0.0  0.2  0.2  0.2  0.2  0.2  0.0   0.0
```

Starting probabilities, Transission probabilities and Emission probabilites, are all shown above.

```
symbols = LETTERS[1:n]
states  = rep("State_",n)
for ( i in 1:n){
  states[i] = paste(states[i],symbols[i],sep="")
}

mod = initHMM(States=symbols, Symbols = symbols, startProbs=startP, transProbs=P, emissionProbs = emiss
```

Question 2

```
set.seed(123)
simulated_path = simHMM(mod,100)

print(simulated_path)
```

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

Question 3

```r
obs = simulated_path$observation

f = forward(mod,obs)

filtered = prop.table(t(exp(f)),margin=1)


b = backward(mod, obs)

smoothed = prop.table(t(exp(f + b)),margin=1)


most_probable_path = viterbi(mod,obs)
```

Question 4

```r
guesses = matrix(nrow=100,ncol=3)
colnames(guesses) = c("filtered", "smoothed", "viterbi")

for (i in 1:100){
  filtered_guess = symbols[ which.max(filtered[i,])]

  smoothed_guess = symbols[ which.max(smoothed[i,])]

  guesses[i,1:2] = c(filtered_guess, smoothed_guess)
}
guesses[,3] = most_probable_path

acc = colMeans(guesses == simulated_path$states)
acc
```

```
## filtered smoothed  viterbi
##     0.53     0.64     0.36
```

Question 5

```r
simulated_path = simHMM(mod,100)
obs = simulated_path$observation

f = forward(mod,obs)

filtered = prop.table(t(exp(f)),margin=1)


b = backward(mod, obs)

smoothed = prop.table(t(exp(f + b)),margin=1)
```

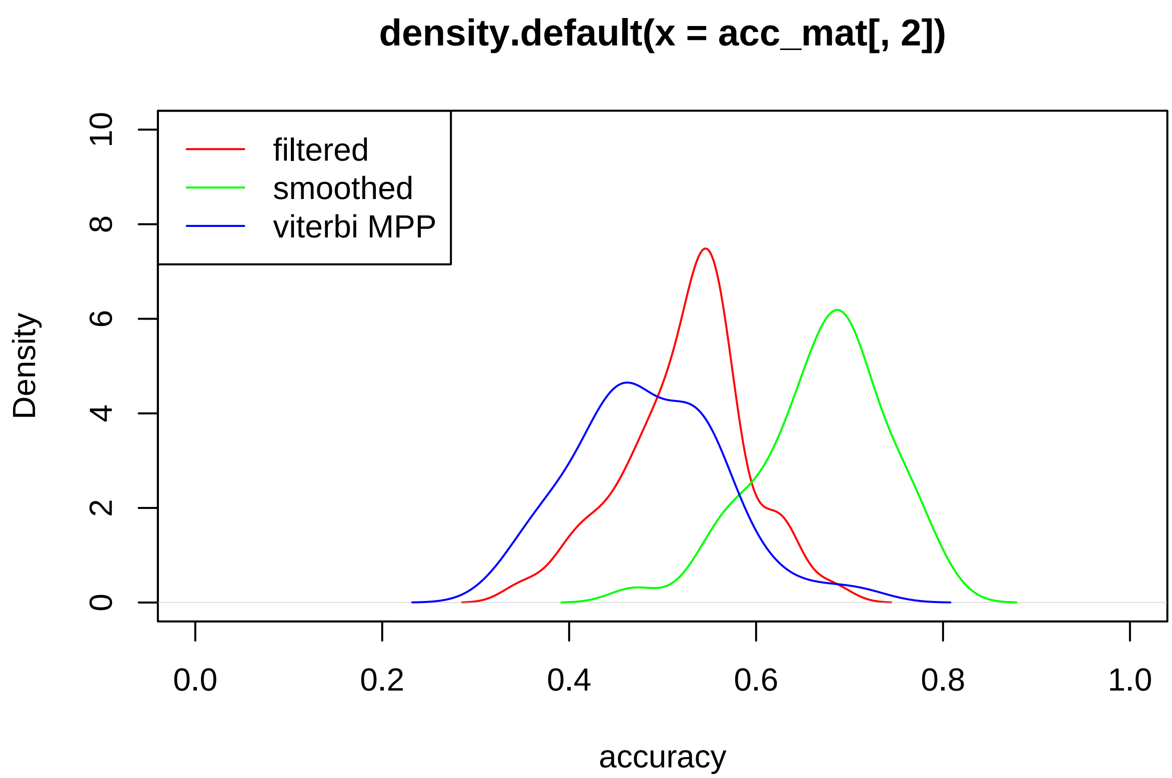
```r
most_probable_path = viterbi(mod,obs)

guesses = matrix(nrow=100,ncol=3)
colnames(guesses) = c("filtered", "smoothed", "viterbi")

for (i in 1:100){
  filtered_guess = symbols[ which.max(filtered[i,])]

  smoothed_guess = symbols[ which.max(smoothed[i,])]

  guesses[i,1:2] = c(filtered_guess, smoothed_guess)
}
guesses[,3] = most_probable_path

acc = colMeans(guesses == simulated_path$states)
acc
```

```
## filtered smoothed  viterbi
##     0.59     0.71     0.48
```

Question 5

```r
acc_mat = matrix(ncol=4, nrow=0)
colnames(acc_mat) = c("NumSteps", "filtered_acc", "smoothed_acc", "viterbi_acc")
for  ( temp in 1:50){
  n = 100

  simulated_path = simHMM(mod,n)
  obs = simulated_path$observation

  f = forward(mod,obs)

  filtered = prop.table(t(exp(f)),margin=1)


  b = backward(mod, obs)

  smoothed = prop.table(t(exp(f + b)) ,margin=1)


  most_probable_path = viterbi(mod,obs)

  guesses = matrix(nrow=n,ncol=3)
  colnames(guesses) = c("filtered", "smoothed", "viterbi")

  for (i in 1:n){
    filtered_guess = symbols[ which.max(filtered[i,])]

    smoothed_guess = symbols[ which.max(smoothed[i,])]
```

```
    guesses[i,1:2] = c(filtered_guess, smoothed_guess)
  }
  guesses[,3] = most_probable_path

  acc = colMeans(guesses == simulated_path$states)
  #print(acc)
  acc_mat = rbind(acc_mat, c(n,acc))


  }


plot(density(acc_mat[,2]), col ="red",xlab="accuracy", xlim=c(0,1), ylim=c(0,10))
lines(density(acc_mat[,3]), col ="green")
lines(density(acc_mat[,4]), col ="blue")

legend("topleft",legend = c("filtered", "smoothed", "viterbi MPP"), col=c("red", "green", "blue"), lty =
```

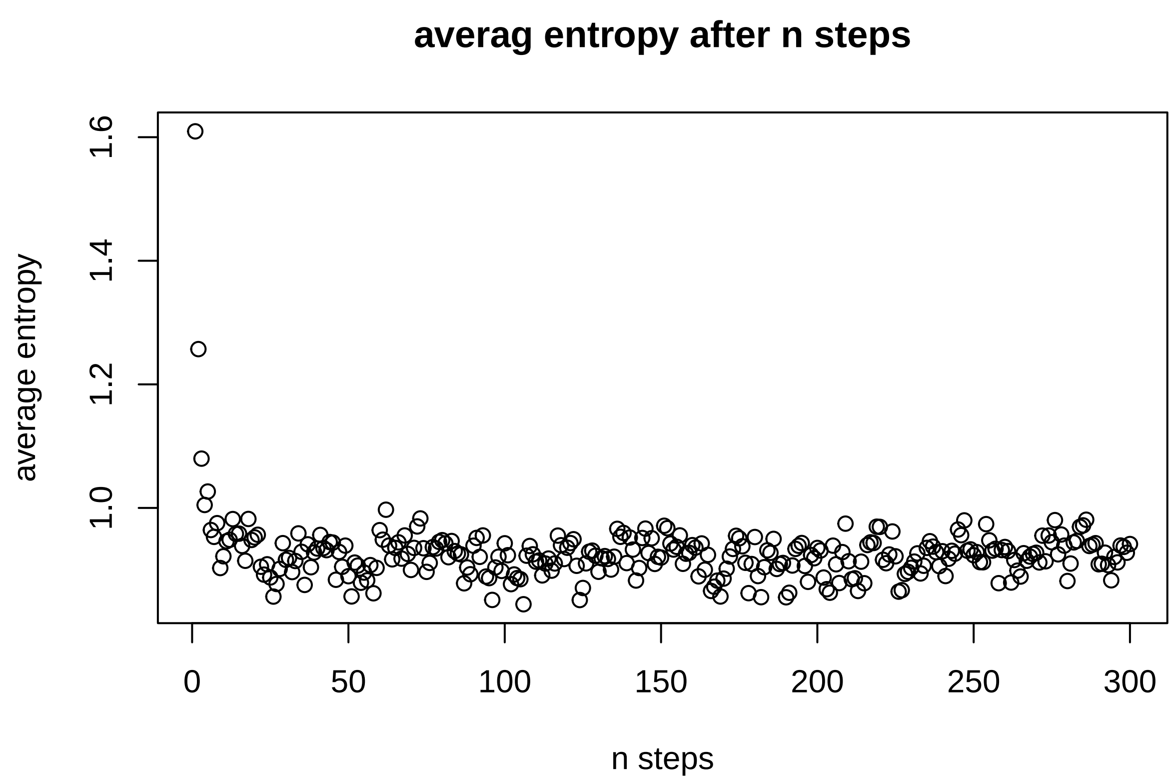


The smoothed distribution takes future data into account and thus has more information avaialbe than the the filtered. This is an "advantage".

The most probable path has the be a valid path which is a restriction not placed on the smoothed distribution.

Question 6

```r
library()
mat_ents = matrix(nrow=300, ncol=150)

for ( i in 1:150){
  simulated_path = simHMM(mod,300)
  obs = simulated_path$observation

  f = forward(mod,obs)

  filtered = prop.table(t(exp(f)),margin=1)
  entropy_vec = apply(filtered,1, entropy.empirical )
  mat_ents[,i] = entropy_vec



}

ent_means = apply(mat_ents,1,mean)
plot(x=1:300, y = ent_means, main="averag entropy after n steps", xlab="n steps", ylab="average entropy
```



There is a sharp decline after the first (~3) steps. After that the values seem the be relatively stable

Question 7

```
set.seed(123)
simulated_path = simHMM(mod,100)
obs = simulated_path$observation

f = forward(mod,obs)

filtered = prop.table(t(exp(f)),margin=1)

prob_101 = round(filtered[100,] %*% P, digits = 3)
rownames(prob_101)= "Inferred Probabilty"
colnames(prob_101) = symbols
prob_101
```

```
##                      A B     C     D     E     F     G H I J
## Inferred Probabilty 0 0 0.049 0.223 0.375 0.277 0.076 0 0 0
```