DAT320_CA3_Exercise2

2022-11-16

```
library(HMM)
library(ggplot2)
```

Exercise2 (a) Compute the following quantities by hand (you may use software for, e.g., computing matrix products or matrix-vector products): *2- and 3-step transition probabilities P(Xt+2|Xt) and P(Xt+3|Xt)* marginal probability distributions of states in t = 2, 3, (P(X2) and P(X3)), *marginal probability distributions of observations in t = 1,2,3, (P(Y1), P(Y2) and P(Y3)) What can we say about marginal probabilities for Yt with higher t?

```
# 2- and 3-step transition probabilities P(Xt+2|Xt) and P(Xt+3|Xt)
transition_matrix <- rbind(c(0.6, 0.4), c(0.1, 0.9))
transition_matrix
```

```
##      [,1] [,2]
## [1,]  0.6  0.4
## [2,]  0.1  0.9
```

```
initial_matrix <- cbind(c(0.5, 0.5))
initial_matrix
```

```
##      [,1]
## [1,]  0.5
## [2,]  0.5
```

```
## P(Xt+2|Xt)
transition_matrix %*% transition_matrix
```

```
##      [,1] [,2]
## [1,] 0.40 0.60
## [2,] 0.15 0.85
```

```
## P(Xt+3|Xt)
transition_matrix %*% transition_matrix %*% transition_matrix
```

```
##       [,1]  [,2]
## [1,] 0.300 0.700
## [2,] 0.175 0.825
```

```
# Marginal probability distributions of states in t = 2, 3, (P(X2) and P(X3))
```

```
## P(X2)
t(transition_matrix %*% transition_matrix) %*% initial_matrix
```

```
##       [,1]
## [1,] 0.275
## [2,] 0.725
```

```
## P(X3)
t(transition_matrix %*% transition_matrix %*% transition_matrix) %*% initial_matrix
```

```
##        [,1]
## [1,] 0.2375
## [2,] 0.7625
```

```
# Marginal probability distributions of observations in t = 1, 2, 3, (P(Y1), P(Y2) and P(Y3))
```

```
## P(Y1)
## P(Yt = 1 | Xt = loaded) * P(Xt = loaded) + P(Yt = 1 | Xt = fair) * P(Xt = fair)
0.1*0.5 + 0.1*0.5
```

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

```
## P(Y2)
## P(Yt = 2 | Xt = loaded) * P(Xt = loaded) + P(Yt = 2 | Xt = fair) * P(Xt = fair)
0.1*0.5 + 0.1*0.5
```

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

```
## P(Y3)
## P(Yt = 3 | Xt = loaded) * P(Xt = loaded) + P(Yt = 3 | Xt = fair) * P(Xt = fair)
0.1*0.5 + 0.1*0.5
```

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

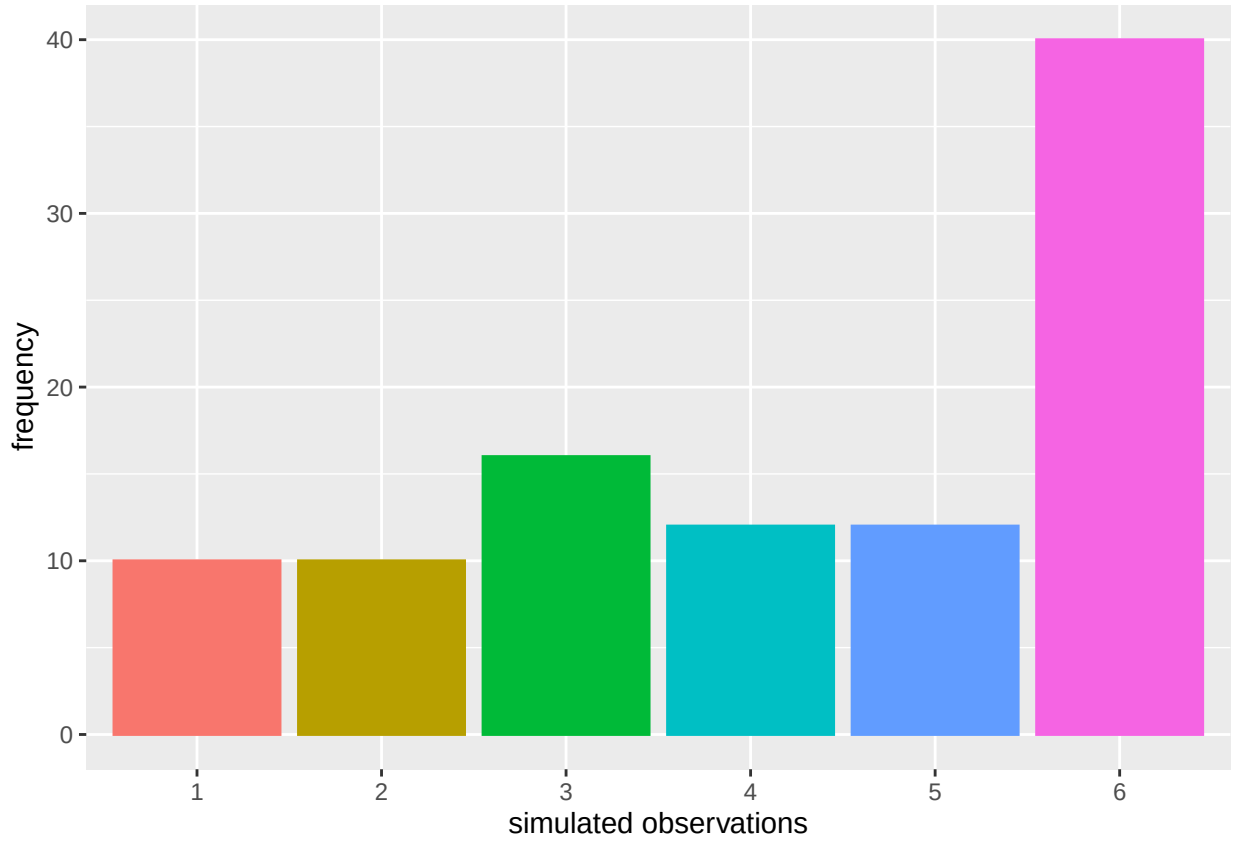
```
# What can we say about marginal probabilities for Yt with higher t?
```

We can say when t equals to 6, the marginal probability P(Y6) = P(Yt = 6 | Xt = loaded) * P(Xt = loaded) + P(Yt = 6 | Xt = fair) * P(Xt = fair) <=> 0.5*0.5 + 0.5*0.5 <=> 0.5 On the other hand, when t is smaller than 6, the marginal probability will be always 0.1 as shown in the former question.

Exercise2 (b) Initialize an HMM in R using the given emission probabilities, initial state proba- bilities, and the transition probabilities. Use function simHMM from package HMM to simulate the scenario in R. In particular, simulate 3 observation and state sequences with length 100, each. For each sequence, plot a histogram of the simulated observations.

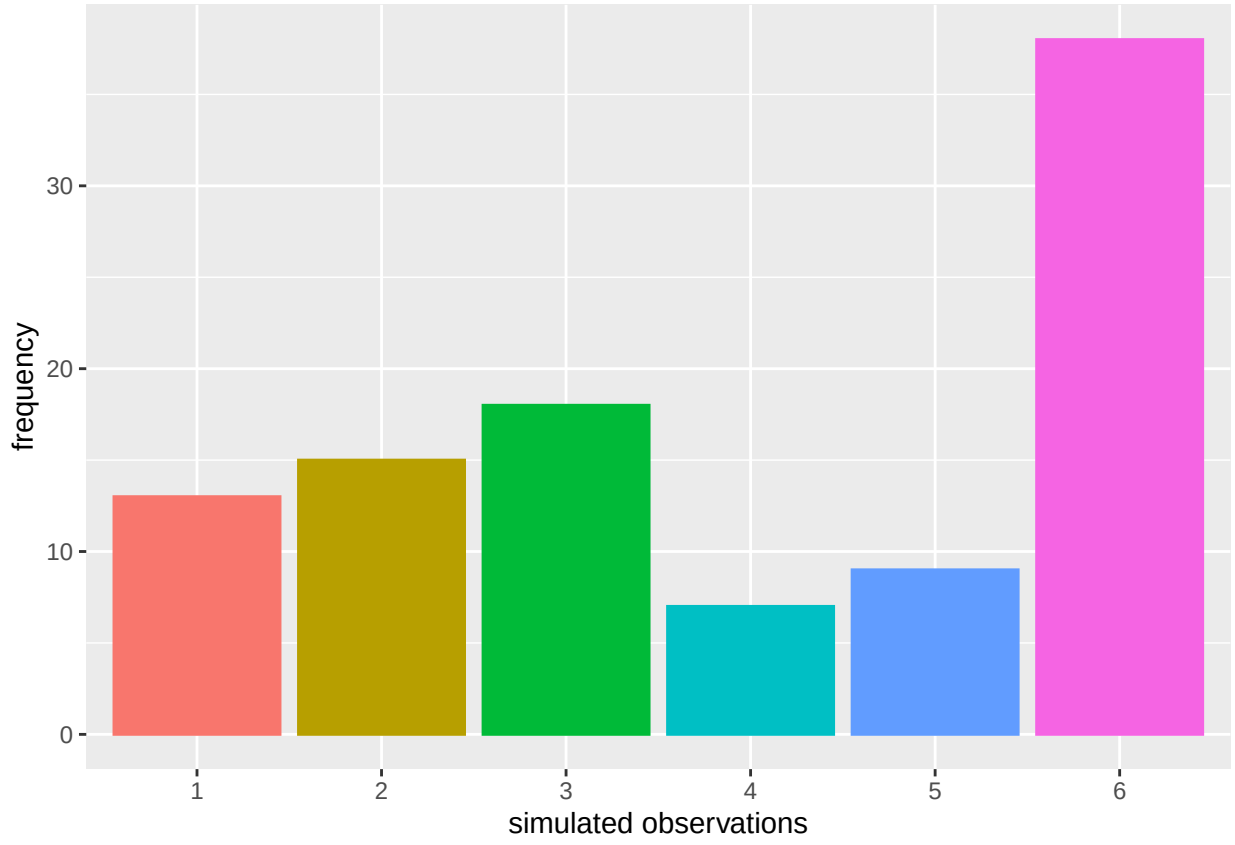
```
# First sequence
set.seed(1)

# initialize HMM (with fixed parameters)
model <- initHMM(
```

```
  c("fair","loaded"), c("1", "2", "3", "4", "5", "6"),
transProbs = rbind(c(0.6, 0.4), c(0.1, 0.9)),
emissionProbs = rbind(c(1/6, 1/6, 1/6, 1/6, 1/6, 1/6), c(1/10, 1/10, 1/10, 1/10, 1/10, 1/2)),
startProbs = c(0.5, 0.5)
)

# simulate time series of length 100 containing observations& states
observations <- HMM::simHMM(model, 100)

# plot a histogram of the simulated observation
seq <- observations$observation
df <- data.frame(seq)
ggplot(df, aes(x = seq, col = seq, fill = seq)) + geom_bar() +  labs(x = "simulated observations", y =
```



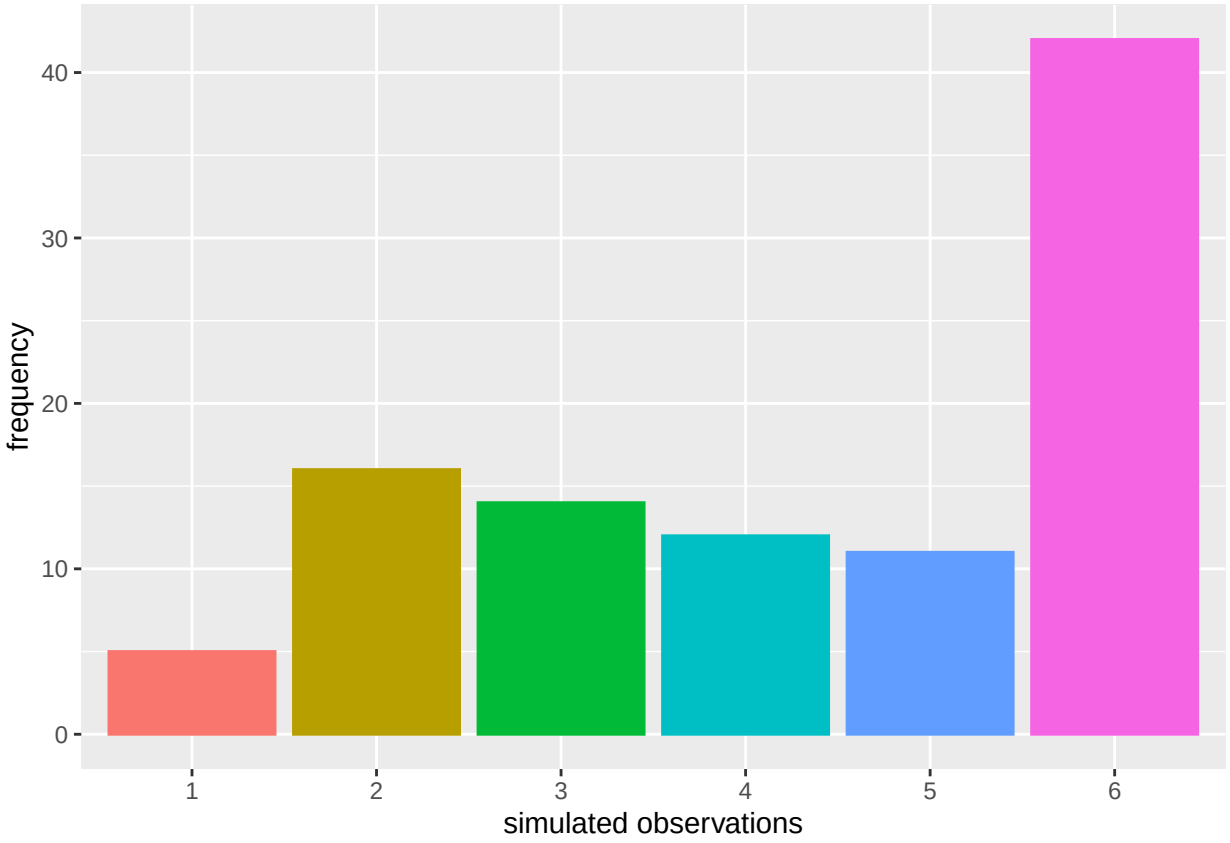

```
# Second sequence
set.seed(2)
# initialize HMM (with fixed parameters)

model <- initHMM(
  c("fair","loaded"), c("1", "2", "3", "4", "5", "6"),
transProbs = rbind(c(0.6, 0.4), c(0.1, 0.9)),
emissionProbs = rbind(c(1/6, 1/6, 1/6, 1/6, 1/6, 1/6), c(1/10, 1/10, 1/10, 1/10, 1/10, 1/2)),
startProbs = c(0.5, 0.5)
)
```

```r
# simulate time series of length 100 containing observations& states
observations <- HMM::simHMM(model, 100)

# plot a histogram of the simulated observation
seq <- observations$observation
df <- data.frame(seq)
ggplot(df, aes(x = seq, col = seq, fill = seq)) + geom_bar() +  labs(x = "simulated observations", y = "
```




```r
# Third sequence
set.seed(3)

# initialize HMM (with fixed parameters)
model <- initHMM(
  c("fair","loaded"), c("1", "2", "3", "4", "5", "6"),
transProbs = rbind(c(0.6, 0.4), c(0.1, 0.9)),
emissionProbs = rbind(c(1/6, 1/6, 1/6, 1/6, 1/6, 1/6), c(1/10, 1/10, 1/10, 1/10, 1/10, 1/2)),
startProbs = c(0.5, 0.5)
)

# simulate time series of length 100 containing observations& states
observations <- HMM::simHMM(model, 100)

# plot a histogram of the simulated observation
seq <- observations$observation
df <- data.frame(seq)
```

```
ggplot(df, aes(x = seq, col = seq, fill = seq)) + geom_bar() +  labs(x = "simulated observations", y =
```



Exercise2 (c) Assume you observe the following sequence of observations yt: 33345116566166143335 Given the HMM parameters specified above, what is the probability that this sequence occurs, if only the fair dice was used? What is the probability if only the loaded dice was used? Which scenario is more likely?

i) Fair dice Since the probability of P(Yt = i|Xt = fair) is always equal to 1/6 no matter the value of i, we can just calculate the P(Yt = 3) and multiply it 20 times to get the answer. P(Yt = 3|Xt = fair) = 1/6 P(Yt = 3) = P(Yt = 3|Xt = fair) * P(Xt = fair) = 1/6 * 1/2 = 1/12

```
fair = (1/12)^ 20
fair
```

```
## [1] 2.608405e-22
```

ii) Loaded dice The probability of P(Yt = i|Xt = loaded) has 2 different values. When i is smaller than 6, the probability is always equal to 1/10, and when i is 6, the probability is 1/2.

a) i < 6 P(Yt = i|Xt = loaded) = 1/10 P(Yt = i) = P(Yt = i|Xt = loaded) * P(Xt = loaded) = 1/10 * 1/2 = 1/20

b) i = 6 P(Yt = i|Xt = loaded) = 1/2 P(Yt = i) = P(Yt = i|Xt = loaded) * P(Xt = loaded) = 1/2 * 1/2 = 1/4

Since there are 15 values that are smaller than 6, and rest are all 6, we can calculate the probability as below:

```
loaded = (1/20)^15 * (1/4)^5
loaded
```

```
## [1] 2.980232e-23
```

When we compare 2 values calculated, we can figure out that the given sequence occurs with higher probability when the fair dice is used.

```
fair > loaded
```

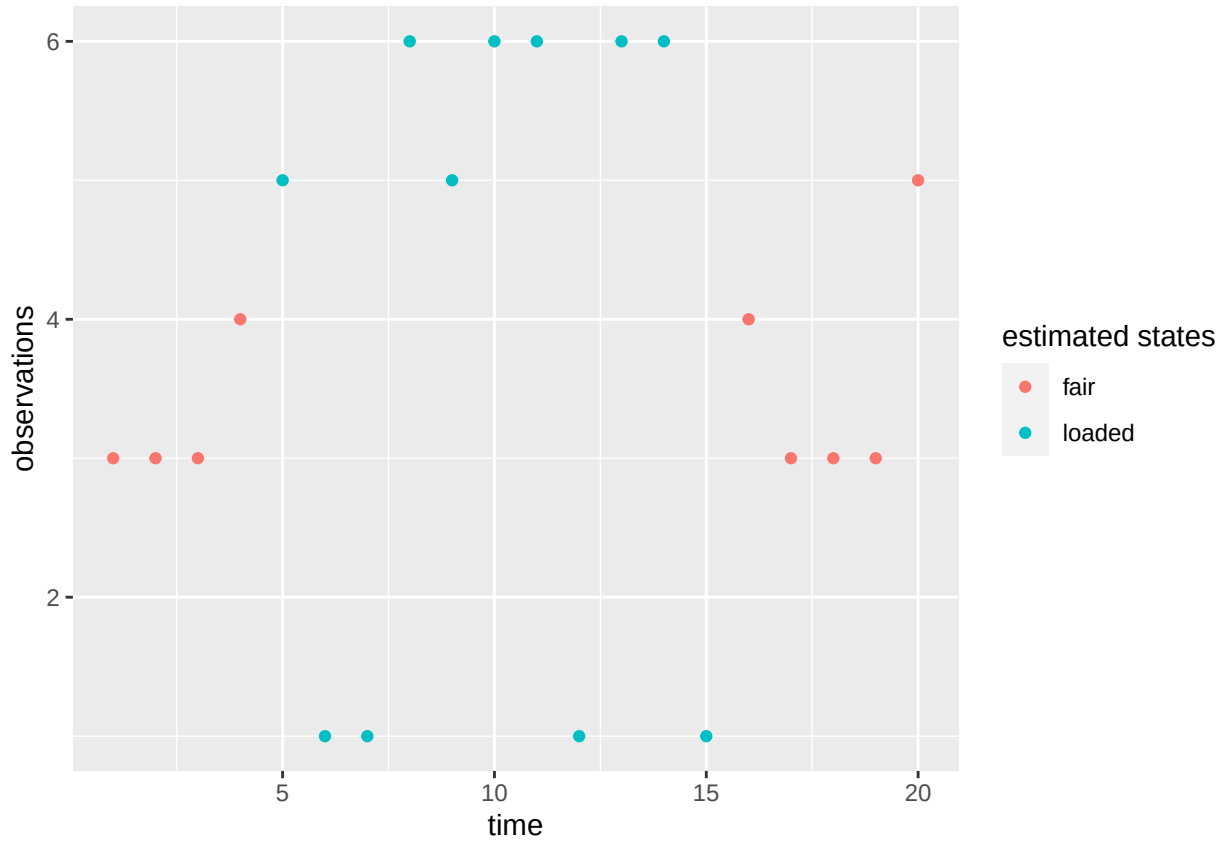
```
## [1] TRUE
```

Exercise2 (d) Use the Baum-Welch algorithm to train the HMM parameters given the observation sequence y1,..., yt in (c). Use the parameters specified in the introduction of the exercise as initialization. Apply the Viterbi algorithm to estimate the state sequence x1,..., xt. Plot the sequence of observations colored by their estimated states. Finally, compute the probability that the observation sequence y1,..., yt occurs given the state sequence x1,..., xt and the original HMM model from the instructions. Compare with the results from (c).

```
set.seed(123)
# Initialize the model
model <- initHMM(
  c("fair","loaded"), c("1", "2", "3", "4", "5", "6"),
transProbs = rbind(c(0.6, 0.4), c(0.1, 0.9)),
emissionProbs = rbind(c(1/6, 1/6, 1/6, 1/6, 1/6, 1/6), c(1/10, 1/10, 1/10, 1/10, 1/10, 1/2)),
startProbs = c(0.5, 0.5)
)

observations_given <- c(3,3,3,4,5,1,1,6,5,6,6,1,6,6,1,4,3,3,3,5)

# Use the BaumWelch algorithm to get the parameters for the given observations
model<- baumWelch(model, observations_given)$hmm

# Apply Viterbi algorithm to estimate the state sequence
hidden_state <- viterbi(model, observations_given)
hidden_state
```

```
##  [1] "fair"   "fair"   "fair"   "fair"   "loaded" "loaded" "loaded" "loaded"
##  [9] "loaded" "loaded" "loaded" "loaded" "loaded" "loaded" "loaded" "fair"
## [17] "fair"   "fair"   "fair"   "fair"
```

```
# Plot the sequence of observations colored by their estimated states
hidden_state <- c(viterbi(model, observations_given))
time <- c (1:20)
df <- data.frame(time, hidden_state, observations_given)
ggplot(data = df) + geom_point(mapping = aes(x = time, y = observations_given, color = hidden_state)) +
```




```
# Compute the prob that the observation sequence occurs given the state sequence and the original HMM m
probability <- c(1/12, 1/12, 1/12, 1/12, 1/20, 1/20, 1/20, 1/4, 1/20, 1/4, 1/4, 1/20, 1/4, 1/4, 1/20, 1/
df <- data.frame(time, hidden_state, observations_given, probability)

answer = 1
for(i in 1:20){
  answer = answer * df$probability[i]
}

answer
```

```
## [1] 2.957256e-21
```

```
fair < answer
```

```
## [1] TRUE
```

```
loaded < answer
```

```
## [1] TRUE
```

```
loaded < fair
```

```
## [1] TRUE
```

We can conclude that the probability for given observations is the highest with the estimated state obtained by Viterbi algorithm, and the probability is the lowest with the loaded dice.