Simulating HMM, basketball game

I. HMM parameters

1. The number of time inerval

t –> the number of time interval that we are observing.

For this case, I split each quarter into 4 parts. In sum, there are 4 quarters and 16 parts.

t = 16

2. The number of hidden states

state –> different kinds of situation on the court.

There are two state for this game simulation, which are leading (1), or falling behind (2).

N = 2

3. The initial distribution

initial –> the initial probability of each hidden state.

Here we assume the team has a slightly better record than their opponent in this season, so they are more likely to be in the leading state rather than falling behind state.

initial = c(0.65, 0.35)

4. The transition probability

transition –> the transition probability

Here we since we assume this team is slightly better, so the transition probability from leading to falling behind is lower
trasition[1,] = c(0.6, 0.4)

And since this team is slightly better, it is more likely that the team change from falling behind to leading
transition[2,] = c(0.7, 0.3)

5. State dependent distribution

state_dist –> discrete, multinominal distribution, which indicates the strategy the team is using.

Here we simply assume there are 3 strategies: aggressive, passive, and neutral.
The probabilities are given based on common senses.

1. If the team is leading, then they are more likely to be passive or neutral, instead of being agressive
   state_dist[1,] = c(0.2, 0.4, 0.4)
2. If the team is falling behind, then they are more likely to be aggrssive. In this case, neutral is less likely but still possible to occur, and passive is almost impossible.
   state_dist[2,] = c(0.6, 0.1, 0.3)

6. The actions produced by state dependent distribution

state_action –> the strategy, 1: active, 2: passive, 3: neutral

Here are the parameters.

```
t <- 16
N <- 2
initial <- c(0.65, 0.35)

transition <- matrix(list(), nrow = 2, ncol = 2)
transition[1, ] <- c(0.6, 0.4)
transition[2, ] <- c(0.7, 0.3)

state_dist <- matrix(list(), nrow = 2, ncol = 3)
state_dist[1, ] = c(0.2, 0.4, 0.4)
state_dist[2, ] = c(0.6, 0.1, 0.3)

state_actions = c(1:3)
```

II. HMM simulation

```
# simulation function
sim_hmm_basketball <- function(t, N, initial, transition, state_dist, state_actions){
  df = data.frame(index = c(1:t), state = numeric(N), observation = numeric(N))
  state = sample(x = c(1:N), size = 1, prob = initial)
  observation = sample(x = state_actions, size = 1, prob = state_dist[state, ])
  df$state[1] <- state
  df$observation[1] <- observation

  for (i in 2:t) {
    trans_prob = transition[state, ]
    state <- sample(x = c(1:N), size = 1, prob = trans_prob)
    observation <- sample(x = state_actions, size = 1, prob = state_dist[state, ])
    df$state[i] <- state
    df$observation[i] <- observation
  }
  return(df)
}
```

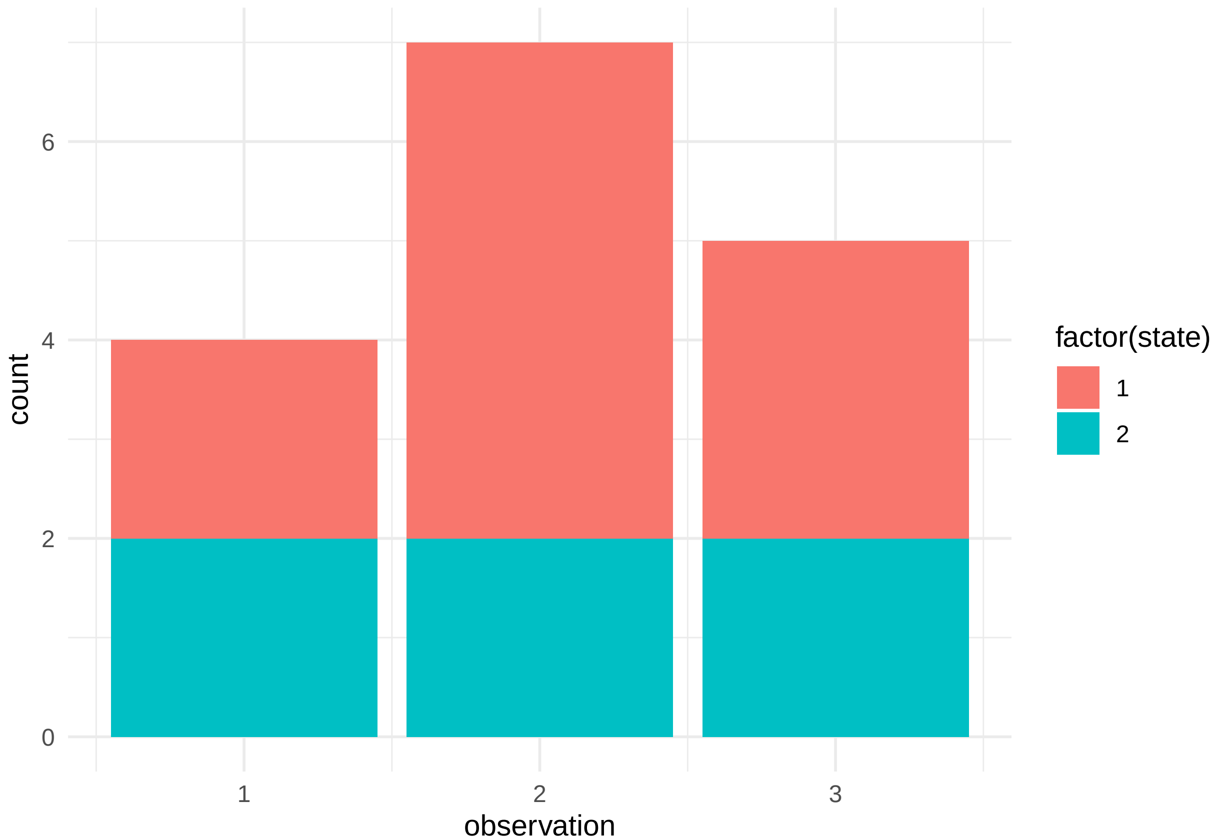
```
# simulation process
set.seed(6)
game1 <- sim_hmm_basketball(16, N, initial, transition, state_dist, state_actions)

knitr::kable(game1)
```

index	state	observation
1	1	1
2	1	2
3	2	2
4	2	3
5	1	2
6	2	2
7	1	2

index	state	observation
8	2	1
9	1	3
10	1	3
11	2	3
12	1	2
13	2	1
14	1	1
15	1	3
16	1	2

Obsevation and State

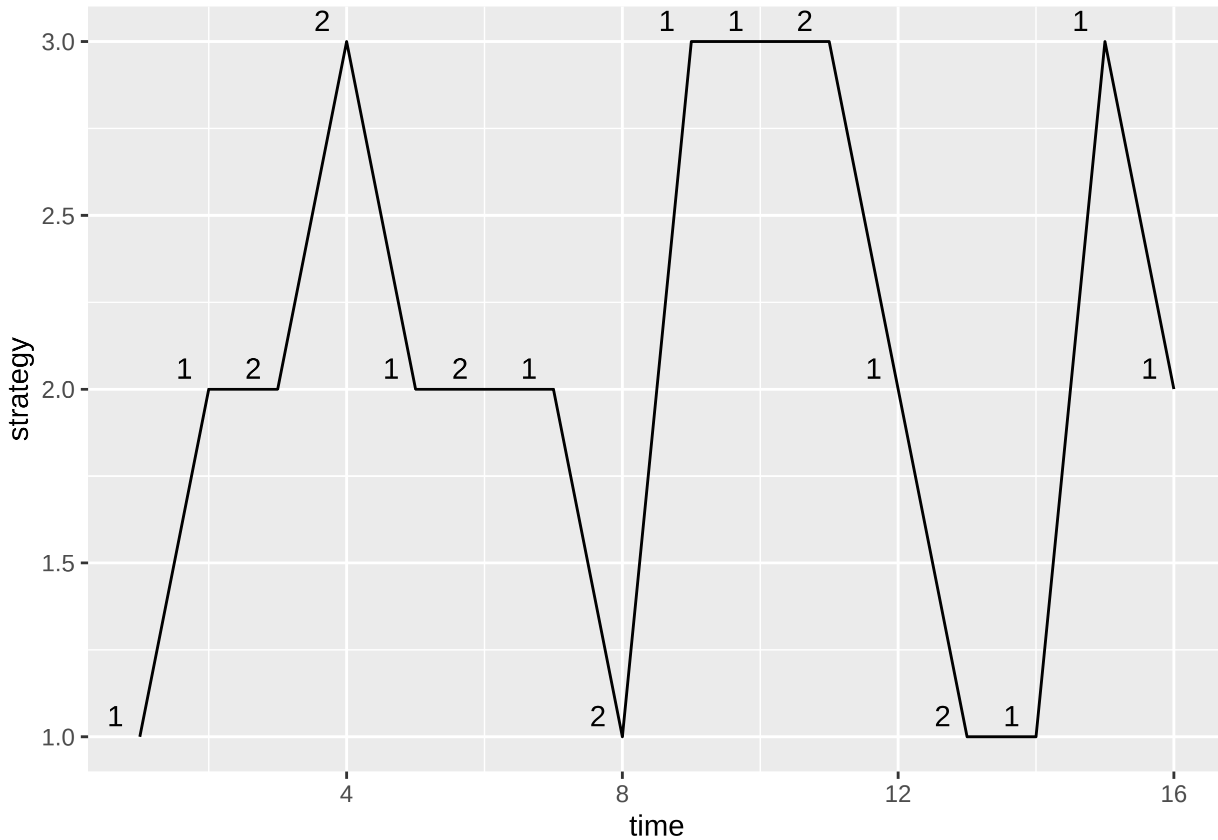
```
ggplot(game1, aes(observation, fill = factor(state))) + geom_bar() + theme_minimal()
```



Remember, state 1 is leading and state 2 is falling behind. Therefore, we can see that when the team is leading (state 1), it is more likely that the team behaves passively, and when the team is falling behind (state 2), it is more likely that the team behaves aggresively. This is corresponding to our assumption.

Strategy change over time and the state condition

```
ggplot(game1, aes(x = index, y = observation)) +
  geom_line() +
  xlab("time") + ylab("strategy") +
  geom_text(aes(label = state, hjust = 2, vjust = -0.5))
```



Considering it as a real game, we can see some interesting facts. For example, one of the simulations I have seen is that, the team is falling behind at the beginning, and their strategies are active-neutral-active, and they eventually switch from falling behind to leading.

Of course, I think the model is problematic, since the next state (i.e. leading or falling behind) should not be independent from the current strategy, but HMM doesn't have the property (i.e. the current result and the next state are independent).

This is just a simple simulation comes into my mind first, I will continue to fit some discrete and continuous distribution as practice.

Simulate HMM for continuous data.

This is a simulation without much complicated assumption or cases.

Simply, there are two hidden states ($z = 1$ and $z = 2$) and the state-dependent distribution are $N(\mu_z, 1)$.

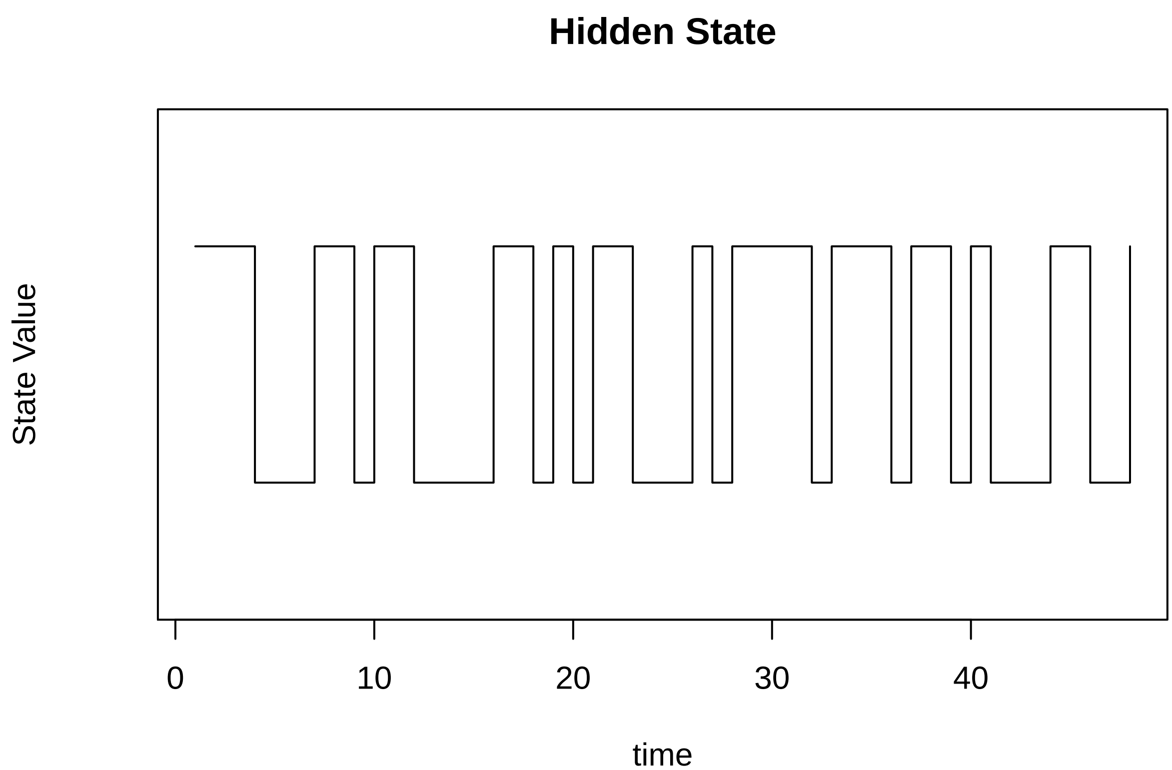
Here, the initial probability of being in any of two states and the probability of transiting between two states are all the same (i.e. all 0.5), and the mean of state dependent distribution to be 4 and 6.

The time we expect here is the same as the number of minutes in a bseketball game (i.e. 48).

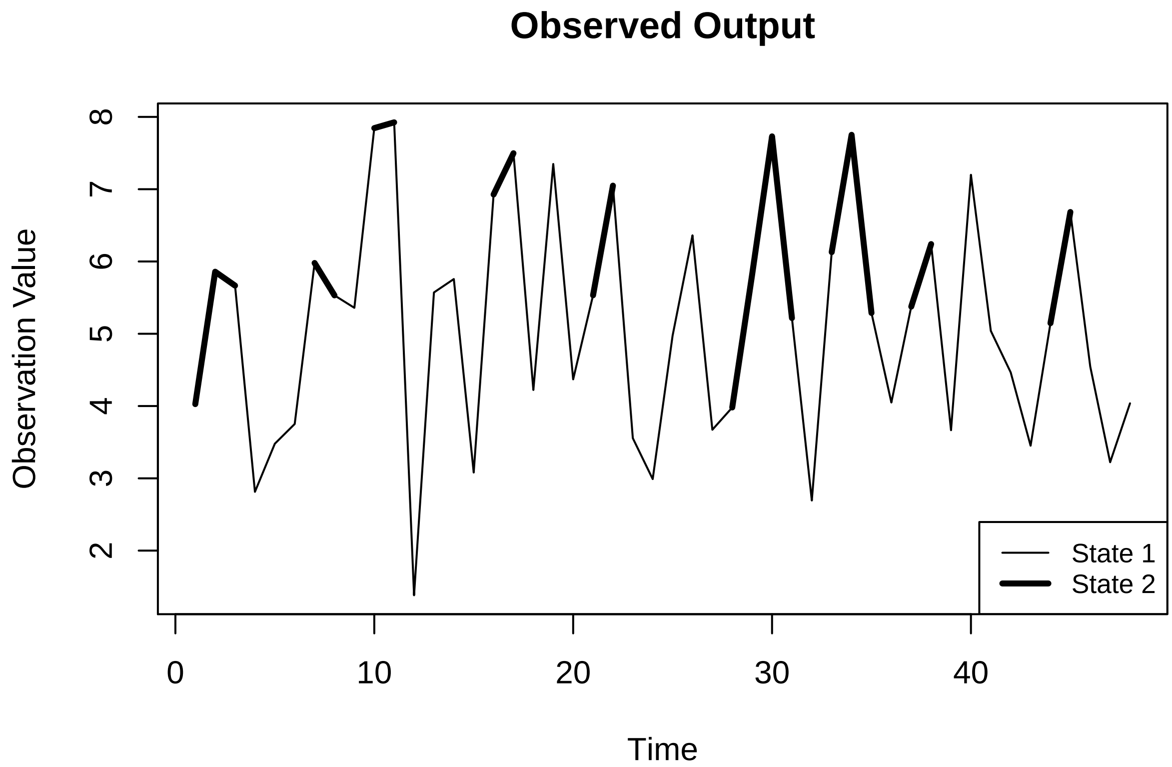
```
t <- 48
N <- 2
initial = c(0.5, 0.5)
transition <- matrix(list(), nrow = 2, ncol = 2)
transition[1, ] = c(0.5, 0.5)
transition[2, ] = c(0.5, 0.5)
state_dist = c(4, 6)
```

```r
sim_hmm_basketball_cont <- function(t, N, initial, transition, state_dist){
  df = data.frame(index = c(1:t), state = numeric(N), observation = numeric(N))
  state = sample(x = c(1:N), size = 1, prob = initial)
  obs = rnorm(n = 1, mean = state_dist[state], sd = 1)
  df$state[1] = state
  df$observation[1] = obs

  for (i in 2:t) {
    trans_prob = transition[state, ]
    state = sample(x = c(1:N), size = 1, prob = trans_prob)
    obs = rnorm(n=1, mean = state_dist[state], sd = 1)
    df$state[i] = state
    df$observation[i] = obs
  }

  return(df)
}
```

```r
set.seed(9)
df = sim_hmm_basketball_cont(t, N, initial, transition, state_dist)
plot(df$state, type = 's', main = 'Hidden State', ylab = 'State Value', xlab = 'time', ylim = c(0.5, 2.
```




```r
plot(df$observation, type = 'l', main = "Observed Output", ylab = 'Observation Value', xlab = 'Time')
y_plt = df$observation
y_plt[df$state == 1] <- NA
lines(y_plt, lwd = 3)
legend("bottomright", c("State 1", "State 2"), lty = c(1, 1), lwd = c(1, 3), cex = 0.8)
```


Observed Output
Observation Value
8
7
6
5
4
3
2
0
10
20
30
40
Time
State 1
State 2