HW2

```r
library(tidyverse)
```

Preliminary

Transition matrix is given:

```
     [,1] [,2] [,3] [,4]
[1,]  0.7  0.0  0.3  0.0
[2,]  0.5  0.0  0.5  0.0
[3,]  0.0  0.4  0.0  0.6
[4,]  0.0  0.2  0.0  0.8
```

To find log run proportions for states 1, 2, 3, 4, we need to solve a system of these equations:

$$\begin{aligned}
\pi_1 &= 0.7 * \pi_1 + 0.5 * \pi_2 \\
\pi_2 &= 0.4 * \pi_3 + 0.2 * \pi_4 \\
\pi_3 &= 0.3 * \pi_1 + 0.5 * \pi_2 \\
\pi_4 &= 0.6 * \pi_3 + 0.8 * \pi_4 \\
1 &= \pi_1 + \pi_2 + \pi_3 + \pi_4
\end{aligned}$$

From the above equations, we obtain the following relations:

$$\begin{aligned}
\pi_1 &= \frac{0.5}{0.3} * \pi_2 \\
\pi_3 &= \pi_2 \\
\pi_4 &= 3 * \pi_3 \\
1 &= \pi_1 + \frac{.3}{.5} * \pi_1 + \frac{.3}{.5} * \pi_1 + 2 * \pi_1
\end{aligned}$$

We obtain $\pi_1 = 0.25$, which leads us to the following solution for the stationary probabilities:

$$\pi_1 = 0.25, \pi_2 = 0.15, \pi_3 = 0.15, \pi_4 = 0.45$$

Simulation

A

First 40 random uniform numbers

```
set.seed(7782)
random_numbers = runif(n = 50000, min = 0, max = 1)
head(random_numbers, 40)
```

```
 [1] 0.57591775 0.38141629 0.14709238 0.58570900 0.22139279 0.66158847
 [7] 0.98655609 0.31203849 0.75027423 0.52559617 0.99721537 0.31371674
[13] 0.51441718 0.06870092 0.78487619 0.10662185 0.78612457 0.89742463
[19] 0.98234323 0.47612389 0.22816976 0.36098049 0.36811226 0.53395252
[25] 0.02113360 0.83705842 0.59746702 0.32605528 0.06787777 0.66174659
[31] 0.24406246 0.50706248 0.88470714 0.22922533 0.88363327 0.52883161
[37] 0.96959248 0.97509250 0.75709546 0.89050376
```

B

Suppose we assign an intiial state according to the rules given below:

```
State = case_when(random_numbers <= 0.25 ~ 1,
                  random_numbers >= 0.25 & random_numbers <= .5 ~ 2,
                  random_numbers >= .5 & random_numbers <= .75 ~ 3,
                  random_numbers >= .75  ~ 4
                  )

State_0 = State[1]
```

Then, the first state is 3. Thus, we will select the next state with probabilities 0, 0.4, 0, 0.6 corresponding to states 1, 2, 3, 4

Using code below, we select the next state according to transition probabilities

```
set.seed(1728)
sample(c(1,2,3,4), size = 1, prob = P[State_0,]) -> selected_state
```

Next selected state, $X_1 = 4$

C

Now we replicate the whole process using a loop. This code chunk follows the procedure that was given to us. At each step, we select X_n as defined by the 50,000 uniform random numbers, and pick the state X_{n+1} using transtion probabilities from the matrix. At the next step we select a new state for X_{n+1}, which might not agree with what was sampled during the transition from X_n to X_{n+1}.

```
res = data.frame(step = seq(from = 1, to = 50000, by = 1),
                 state = NA)

# We have transition matrix P

set.seed(78916)

for(i in 1:nrow(res)){

  if(i %% 100 == 0){print(i)}

  current_state = State[i]
  next_state = sample(c(1,2,3,4), size = 1, prob = P[current_state,])

  res$step[i] = i
  res$state[i] = next_state

}
```

```r
write.csv(res, "res1.csv")
```

For comparison, I generated a Markov Chain Process that would use a state selected in a transiton from X_n to X_{n+1} for a stansition from X_{n+1} to X_{n+2}. I selected an intiial state from the fist randomly generated uniqform number. Code for this procedure is also given below:

```r
res = data.frame(step = seq(from = 1, to = 50000, by = 1),
                 state = NA)

# We have transition matrix P

set.seed(78916)
next_state = State[1]

for(i in 1:nrow(res)){

  if(i %% 100 == 0){print(i)}

  next_state = sample(c(1,2,3,4), size = 1, prob = P[next_state,])

  res$step[i] = i
  res$state[i] = next_state

}

write.csv(res, "res2.csv")
```

D

Using simulated random values, we can show that he long term proportions from the process described in (C) produces the following data structure:

```r
head(desired)[,-1]
```

```
# A tibble: 6 x 2
   step state
  <dbl> <dbl>
1     1     3
2     2     1
3     3     4
```

```
4      4      3
5      5      1
6      6      1
```

Long running proportions of state are given below:

```
table(desired$state)/nrow(desired)
```

```
      1       2       3       4 
0.30504 0.14986 0.19960 0.34550 
```

The difference between long run proportions and hand-solution is given below:

```
table(desired$state)/nrow(desired) - c(0.25, 0.15, 0.15, 0.45)
```

```
       1        2        3        4 
 0.05504 -0.00014  0.04960 -0.10450 
```

These proportions disagree with the solution we calculated by hand, which makes sense, because at each step we modified the process and use auxiliary information to make transitions between states.

For comparison, I also calculated long run proportions from process that relied solely on the initial state from the uniform random numebrs, but used transition states and transition probabilities to make the decision on the next state.

Results are given below:

```
table(correct$state)/nrow(correct)
```

```
      1       2       3       4 
0.25116 0.15028 0.15030 0.44826 
```

```
table(correct$state)/nrow(correct) - c(0.25, 0.15, 0.15, 0.45)
```

```
       1        2        3        4
 0.00116  0.00028  0.00030 -0.00174
```

We are now within a Monte Carlo error away from the true long running proportions.

E