ETC3430

Financial Mathematics under Uncertainty

Assignment 1, Semester 1, 2022

# Question 1

Perform the following simulation exercise in Matlab or R, using your student number as the seed( rng( your student no.)). Consider a three state Markov Chain(“sunny”, “rainy” and “overcast”) to model the daily weather in Melbourne.

1. Without using data, suggest a reasonable estimate of the transition matrix. The later questions should be considered based on your estimates.
2. Given each starting state respectively, what is the most likely state of the weather in the next five days. Present your answer in a table.   
   Table

   Description automatically generated

Table

Description automatically generated

1. Label the three states as , i.e., if the weather is sunny, if the weather is rainy and if the weather is overcast. Given each starting state respectively, what is in the next five days. Present your answer in a table.  
     
   Table

   Description automatically generated
2. Compare and contrast your answers from (b) and (c)  
   Part b shows the mode given that the previous state at each given day, while part c is showing the expected value given that the previous state at each given day.
3. Generate histograms of with an initial distribution of generated with a seed given by your postcode for sample sizes , compare them with the associated probability density function of , then comment on your findings.

set.seed(3149)

P <- matrix(

    data = c(0.45, 0.35, 0.20, 0.36, 0.34, 0.30, 0.25, 0.65, 0.10),

    nrow = 3, ncol = 3, byrow = TRUE

)

# Cumulative Probability

CP <- t(apply(P, 1, cumsum))

repeater <- function(sim\_size) {

    X <- matrix(0, sim\_size + 1, 1)

    X[1, 1] <- 1

    for (day in 1:3) {

        for (i in 1:sim\_size) {

            u <- runif(1)

            X[i + 1, 1] <- 1 \* (u < CP[X[i, 1], 1]) +

                2 \* (u < CP[X[i, 1], 2]) \* (u > CP[X[i, 1], 1]) +

                3 \* (u > CP[X[i, 1], 2])

        }

        g <- X %>%

            as\_tibble() %>%

            ggplot(aes(x = V1)) +

            geom\_histogram() +

            labs(

                title = paste("Day ", day),

                subtitle = paste("n =  ", sim\_size),

                x = "State"

            )

        print(g)

    }

}

sims <- c(100, 1000, 10000, 100000)

for (index in 1:4) {

    repeater(sims[index])

}

Chart, histogram

Description automatically generated

Chart, histogram

Description automatically generatedChart

Description automatically generated with medium confidence

Chart

Description automatically generated

From the simulation of , the values converge on the pdf value for since the accuracy of the models increases as more datapoints are tested.

1. Use samples with sizes to estimate , comment on the Monte-Carlo estimation with the theoretical answer as increases

set.seed(NULL)

P <- matrix(

    data = c(0.45, 0.35, 0.20, 0.36, 0.34, 0.30, 0.25, 0.65, 0.10), # these are my vals so pls change lol

    nrow = 3, ncol = 3, byrow = TRUE

)

# Cumulative Probability

CP <- t(apply(P, 1, cumsum))

markov\_proc\_del\_start <- function(sim\_size, starting\_day) {

    X <- matrix(1, sim\_size + 1, 1)

    for (day in starting\_day:3) {

        for (i in 1:sim\_size) {

            u <- runif(1)

            X[i, 1] <- 1 \* (u < CP[X[i, 1], 1]) +

                2 \* (u < CP[X[i, 1], 2]) \* (u > CP[X[i, 1], 1]) +

                3 \* (u > CP[X[i, 1], 2])

        }

    }

    pi <- matrix(0, 1, 3)

    pi[1, 1] <- sum(X == 1) / sim\_size

    pi[1, 2] <- sum(X == 2) / sim\_size

    pi[1, 3] <- sum(X == 3) / sim\_size

    return(pi)

}

sims <- c(100, 1000, 10000, 100000)

for (index in 1:4) {

    print(glue("Simulation size: {sims[index]}"))

    print(markov\_proc\_del\_start(sims[index], 2))

    cat("\n")

    cat("\n")

}

theory <- c(1, 0, 0) %\*% (P %^% 2)

print(glue("Theoretical Value:"))

print(theory)

**Simulation size: 100**

[,1] [,2] [,3]

[1,] 0.32 0.43 0.26

**Simulation size: 1000**

[,1] [,2] [,3]

[1,] 0.393 0.383 0.225

**Simulation size: 10000**

[,1] [,2] [,3]

[1,] 0.3718 0.4172 0.2111

**Simulation size: 1e+05**

[,1] [,2] [,3]

[1,] 0.37707 0.40805 0.21489

**Theoretical Value:**

[,1] [,2] [,3]

[1,] 0.3785 0.4065 0.215

Reasoning:

The Monte Carlo estimation is done through random sampling to estimate the true probability. However for this to be done accurately, the sampling size needs to be large, since bias in the sample need to minimised. This can be seen in the different values of n that is used to estimate, . As n grows so does the accuracy, it converges on the theoretical value.

1. Use samples with sizes to estimate , comment on the Monte-Carlo estimation with the theoretical answer as increases.

Markov processes are memoryless hence, .

sims <- c(100, 1000, 10000, 100000)

for (index in 1:4) {

    print(glue("Simulation size: {sims[index]}"))

    print(markov\_proc\_del\_start(sims[index], 3))

    cat("\n")

    cat("\n")

}

theory <- c(1, 0, 0) %\*% (P %^% 1)

print(glue("Theoretical Value:"))

print(theory)

**Simulation size: 100**

[,1] [,2] [,3]

[1,] 0.44 0.33 0.24

**Simulation size: 1000**

[,1] [,2] [,3]

[1,] 0.447 0.36 0.194

**Simulation size: 10000**

[,1] [,2] [,3]

[1,] 0.4444 0.3507 0.205

**Simulation size: 1e+05**

[,1] [,2] [,3]

[1,] 0.44935 0.35178 0.19888

**Theoretical Value:**

[,1] [,2] [,3]

[1,] 0.45 0.35 0.2

Markov processes are memoryless hence, . The Monte Carlo estimation is done through random sampling to estimate the true probability. However for this to be done accurately, the sampling size needs to be large, since bias in the sample need to minimised. This can be seen in the different values of n that is used to estimate, . As n grows so does the accuracy, it converges on the theoretical value.

1. Compare and contrast the results from (f) and (g).