

STAT 325 – Handout 12

Introduction to Markov Chains (4.1, 4.2)

Example 12-1: Lunch

Suppose that every day I have either a taco or a slice of pizza for lunch. The probabilities depend on what I had the previous day. If I get a slice of pizza on one day, then I switch to a taco the next day with probability .8 and I have pizza again with probability .2. If I get a taco on one day, then I have a taco again the next day with probability .4 and I switch to pizza with probability .6. If this is really how I make my lunch decision every day, what will happen in the long run? How often will I have a taco, and how often will I have pizza?

a) Which do you think I'll have more often in the long run? How often do you think I'll make that choice?

We'll soon learn how to solve this problem analytically, but let's first use simulation to produce an approximate solution. Here is some R code to simulate this lunch process, with the first lunch choice equally likely to be pizza or a taco and future choices made according to the probabilities above:

```
N = 100000
lunch = rep(NA, times = N)
rand = runif(N,0,1)
if (rand[1] < .5) {lunch[1] = "taco"} else {lunch[1] = "pizza"}
for (i in 2:N) {
  if ((lunch[i-1] == "taco") & (rand[i] < .6)) {lunch[i] = "pizza"}
  if ((lunch[i-1] == "taco") & (rand[i] >= .6)) {lunch[i] = "taco"}
  if ((lunch[i-1] == "pizza") & (rand[i] < .8)) {lunch[i] = "taco"}
  if ((lunch[i-1] == "pizza") & (rand[i] >= .8)) {lunch[i] = "pizza"}
}
table(lunch)
```

b) Execute this code, and report how many of the 100,000 simulated lunches were a taco and how many were pizza. Also report approximate probabilities for each lunch option.

- A **stochastic process** is a collection of random variables indexed by natural numbers, denoted by $\{X(n)\}$ for non-negative integers n .
 - The possible values/outcomes of $X(n)$ are called the **states** of the stochastic process, denoted by $s(n)$.
- A stochastic process satisfies the **Markov property** if the probabilities of states at any point in time depend only on the state at the previous point in time, not on earlier states.
 - Mathematically, the Markov property says that $\Pr[X(n+1) = s(n+1) \mid X(n) = s(n), X(n-1) = s(n-1), \dots, X(1) = s(1), X(0) = s(0)] = \Pr[X(n+1) = s(n+1) \mid X(n) = s(n)]$.
- A stochastic process that satisfies the Markov property is called a **Markov chain**.

c) Is this lunch process a Markov chain? Explain why or why not. If it is, identify its states.

- The **one-step transition probability** of a Markov chain is the conditional probability $\Pr[X(n+1) = j \mid X(n) = i]$.
 - If these transition probabilities do not depend on n , the process is said to be **time-homogeneous** or *stationary*.
 - These probabilities are often organized into a **one-step transition matrix**.
 - Probabilities sum to 1 across *rows* of this matrix.

d) Organize the given conditional probabilities into a one-step transition matrix.

Example 12-2: Ping-pong tournament

Suppose that three ping-pong players engage in a sequence of games. Only two people can play at a time, so they adopt the following rotation policy: Whoever loses the current game steps aside for the next game, allowing the winner of the current game and the player currently sitting out to play in the next game. Suppose that player A is the best and has a .7 probability of beating B and a .8 probability of beating C in a game. Also suppose that B has a .6 probability of beating C in a game. Assume that results of all games are independent of each other.

- a) What are the three possible match-ups of players in this situation?
- b) After A plays B, what are the possibilities for the next match? What are the probabilities?
- c) After A plays C, what are the possibilities for the next match? What are the probabilities?
- d) After B plays C, what are the possibilities for the next match? What are the probabilities?
- e) What are the states of this stochastic process?
- f) Does the Markov property hold here? Explain why or why not.
- g) Report the transition probability matrix for this process.

Suppose that this sequence of games continues for a long time. Some interesting questions are:

- How often will each matchup be played in the long run?
- How often will each person play in the long run?
- How often will each player win in the long run?
- What is each player's probability of winning a game that they play in this process?

Again we will soon learn how to answer these questions analytically, but for now we will rely on simulation to produce approximate solutions. Our course website will have a link to the following R code for conducting this simulation:

```
# start with N = number of repetitions, pAB = Pr(A beats B), pAC = Pr(A beats C), pBC = Pr(B beats C)
#
matchup = rep(NA, times = N)
winner = rep(NA, times = N)
loser = rep(NA, times = N)
#
# initial matchup
rand1 = runif(1,0,1)
if (rand1 < 1/3) {matchup[1] = "AB"}
if (rand1 >= 1/3 & rand1 < 2/3) {matchup[1]="AC"}
if (rand1 >= 2/3) {matchup[1] = "BC"}
#
# generate random numbers to determine winners
rand = runif(N,0,1)
#
# conduct loop
for (i in 1:N) {
  if ((matchup[i] == "AB") & (rand[i] < pAB)) {winner[i] = "A"; loser[i] = "B"; matchup[i+1] = "AC"}
  if ((matchup[i] == "AB") & (rand[i] >= pAB)) {winner[i] = "B"; loser[i] = "A"; matchup[i+1] = "BC"}
  if ((matchup[i] == "AC") & (rand[i] < pAC)) {winner[i] = "A"; loser[i] = "C"; matchup[i+1] = "AB"}
  if ((matchup[i] == "AC") & (rand[i] >= pAC)) {winner[i] = "C"; loser[i] = "A"; matchup[i+1] = "BC"}
  if ((matchup[i] == "BC") & (rand[i] < pBC)) {winner[i] = "B"; loser[i] = "C"; matchup[i+1] = "AB"}
  if ((matchup[i] == "BC") & (rand[i] >= pBC)) {winner[i] = "C"; loser[i] = "B"; matchup[i+1] = "AC"}
}
table(matchup)
table(winner)
table(loser)
#
games = table(winner) + table(loser)
propwin = table(winner)/games
proplose = table(loser)/games
propwin; proplose
```

h) Run this code to simulate 10,000 games of this process. Use the results to produce approximate answers for all of the questions above.

