

PSTAT160A Stochastic Processes

Section 7

Authors: Denisse Alejandra Escobar Parra, John Inston

Date: November 25, 2025

Problem 1 - Dobrow 5.3

Commuters in an urban center either drive by car, take the bus, or bike to work. Based on recent changes to local transportation systems, urban planners predict yearly behavior changes of commuters based on a Markov model $(X_n)_{n \geq 1}$ with transition matrix

$$P = \begin{pmatrix} 0.7 & 0.2 & 0.1 \\ 0.1 & 0.8 & 0.1 \\ 0.05 & 0.15 & 0.8 \end{pmatrix}.$$

1. In a city of 10,000 commuters, 6,000 currently drive cars, 3,000 take the bus, and 1,000 bike to work. Two years after transportation changes are made, how many commuters will use each type of transportation? Over the long term, how many commuters will use each type of transportation?
 - Car: 271
 - Bus: 101
 - Bike: 21
2. The European Cyclists Federation reports the following levels of CO₂ emissions (in grams) per passenger per kilometer traveled:
 - Car: 271
 - Bus: 101
 - Bike: 21What is the current average amount of CO₂ emissions per kilometer traveled? How does the average change over the long term?

Solution

(1) After two years, the number of commuters will be

$$(6000, 3000, 1000)P^2 = (3645, 4165, 2190)$$

For the long term, we need to find the stationary distribution by solving

$$\pi P = \pi \quad \text{and} \quad \pi_1 + \pi_2 + \pi_3 = 1$$

Solving the linear system we get

$$\pi = \left(\frac{5}{24}, \frac{11}{24}, \frac{1}{3} \right)$$

Therefore, out of 10,000 commuters in the long run, roughly 2083 will drive cars, 4165 take the bus and 2190 will ride a bike.

(2) The current average is

$$\frac{6000 \cdot 271 + 3000 \cdot 101 + 1000 \cdot 21}{10,000} = 195\text{g}$$

On the other hand, notice that the Markov Chain is ergodic. Hence, the long term average can be calculated as

$$\lim_{n \rightarrow \infty} \frac{1}{n} \sum_{i=1}^n h(X_n) = \frac{5}{24} \cdot 271 + \frac{11}{24} \cdot 101 + \frac{1}{3} \cdot 21 = 109.75\text{g}.$$

Problem 2 - Dobrow 5.5

Exhibit a Metropolis–Hastings algorithm to sample from the distribution:

Value	1	2	3	4	5	6
Probability	0.01	0.39	0.11	0.18	0.26	0.05

Use a proposal distribution based on one fair die roll.

Solution

We want to sample from the target distribution on 1, 2, 3, 4, 5, 6 given by

$$\pi = (\pi_1, \pi_2, \pi_3, \pi_4, \pi_5, \pi_6) = (0.01, 0.39, 0.11, 0.18, 0.26, 0.05).$$

We use as proposal a Markov chain T based on one fair die roll, that is,

$$T_{i,j} = \mathbb{P}(Y = j \mid X = i) = \frac{1}{6}, \quad i, j \in \{1, \dots, 6\}.$$

This proposal is symmetric, as we have that

$$T_{i,j} = T_{j,i} \quad \text{for all } i, j.$$

Now, we will follow the MH algorithm:

Step 0. Choose an initial state $X_0 \in \{1, \dots, 6\}$.

Step 1. Given $X_n = i$, we choose a new state j based on $T_{i,j}$, i.e., we roll a fair die and take its value as the proposal.

Step 2. Compute the acceptance ratio

$$a_{i,j} = \frac{\pi_j T_{j,i}}{\pi_i T_{i,j}} = \frac{\pi_j}{\pi_i},$$

by symmetry.

Step 3. Let $U_{n+1} \sim U(0, 1)$ and set

$$X_{n+1} = \begin{cases} j, & \text{if } U_{n+1} \leq a_{i,j}, \\ i, & \text{otherwise.} \end{cases}$$

We repeat Steps 1 through 3. The stationary distribution of X_n is π .

Now, we will implement the algorithm. Notice that we will discard the first few samples to eliminate the bias caused by choosing the arbitrary initial value.

```
# Target distribution
pi_vals <- c(0.01, 0.39, 0.11, 0.18, 0.26, 0.05) # states 1,...,6

# Proposal T(i,j): one fair die roll (independent of i)
T <- function(i){
  # Draw J from a uniform
  j <- sample(1:6, size = 1)
  return(j)
}

# Acceptance ratio a(i,j)
a_ij <- function(i, j){
  pi_vals[j] / pi_vals[i]
}

# Metropolis-Hastings step
MH_step <- function(i){
  j <- T(i)                      # Step 1: propose new state j
  u <- runif(1)                   # U ~ Uniform(0,1)
  a <- a_ij(i, j)                 # Step 2: acceptance ratio

  if (u <= a) {
    return(j)                      # accept
  } else {
    return(i)                      # reject, stay at i
  }
}

# Now we run the Markov chain
set.seed(160)
nsim <- 101000
mcsample <- numeric(nsim)
mcsample[1] <- 1                  # Step 0: initial value X_0

for (n in 1:(nsim - 1)) {
  mcsample[n + 1] <- MH_step(mcsample[n])
```

```

}

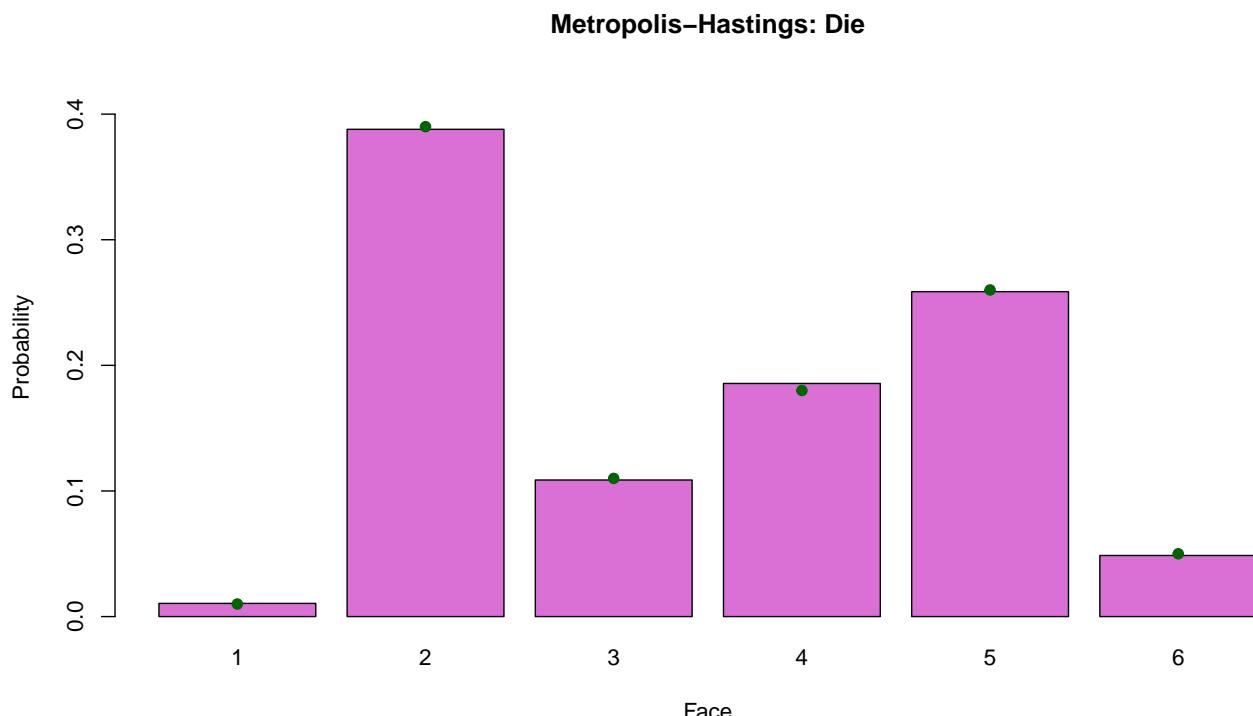
# Burn-in
burn_in <- 1000
post <- mcsample[(burn_in + 1):nsim] # Throw away first samples

# Empirical distribution of the chain
emp_freq <- table(post) / length(post)

# Plot: empirical vs. target probabilities
par(mar = c(6, 4, 4, 2), xpd = NA)
centers <- barplot(emp_freq,
                     main = "Metropolis-Hastings: Die",
                     xlab = "Face", ylab = "Probability",
                     col = "orchid",
                     ylim = c(0, max(c(emp_freq, pi_vals)) * 1.1))

points(centers, pi_vals, pch = 19, col = "darkgreen")

```



```

#In case we'd like to see the results, rather than a graph
results<-rbind(emp_freq,pi_vals)
results

```

	1	2	3	4	5	6
emp_freq	0.0105	0.3879	0.10872	0.18559	0.25866	0.04863
pi_vals	0.0100	0.3900	0.11000	0.18000	0.26000	0.05000

Problem 3 - Dobrow 5.7

Exhibit a Metropolis–Hastings algorithm to sample from a binomial distribution with parameters n and p . Use a proposal distribution that is uniform on $\{0, 1, \dots, n\}$.

Solution

We want to sample from the target distribution on $0, 1, \dots, n$ given by the binomial law with parameters n and p :

$$\pi_k = \mathbb{P}(X = k) = \binom{n}{k} p^k (1-p)^{n-k}, \quad k = 0, 1, \dots, n.$$

We use the proposal T that is uniform on $0, 1, \dots, n$, independent of the current state, i.e.

$$T_{i,j} = \mathbb{P}(Y = j | X = i) = \frac{1}{n+1}, \quad i, j \in 0, \dots, n.$$

This proposal is symmetric, as we have that

$$T_{i,j} = T_{j,i} \quad \text{for all } i, j.$$

Now, we will follow the MH algorithm:

Step 0. Choose an initial state $X_0 \in 0, \dots, n$.

Step 1. Given $X_n = i$, we choose a new state j based on $T_{i,j}$, that is, we sample j uniformly from $0, \dots, n$.

Step 2. Compute the acceptance ratio

$$a_{i,j} = \frac{\pi_j T_{j,i}}{\pi_i T_{i,j}} = \frac{\pi_j}{\pi_i} = \frac{\binom{n}{j} p^j (1-p)^{n-j}}{\binom{n}{i} p^i (1-p)^{n-i}} = \frac{i!(n-i)!}{j!(n-j)!} \left(\frac{p}{1-p} \right)^{j-i}$$

Step 3. Let $U_{n+1} \sim U(0, 1)$ and set

$$X_{n+1} = \begin{cases} j, & \text{if } U_{n+1} \leq a_{i,j} \\ i, & \text{otherwise.} \end{cases}$$

We repeat Steps 1 through 3. The stationary distribution of X_n is the binomial distribution with parameters n and p .

Now, we will implement the algorithm. Notice that we will discard the first few samples to eliminate the bias caused by choosing the arbitrary initial value.

```

# Parameters
n <- 10
p <- 0.3

states <- 0:n

# Target probabilities pi_k = P(X = k)
pi_vals <- dbinom(states, size = n, prob = p)

# Proposal T(i,j): uniform over {0,...,n}, independent of i
T <- function(i){
  j <- sample(states, size = 1)
  return(j)
}

# Acceptance ratio a(i,j)
a_ij <- function(i, j){
  # i, j in {0,...,n}; BUT pi_vals is indexed from 1,...,n+1
  pi_vals[j + 1] / pi_vals[i + 1]
}

# Metropolis-Hastings step
MH_step <- function(i){
  j <- T(i)                      # Step 1: propose new state j
  u <- runif(1)                   # U ~ Uniform(0,1)
  a <- a_ij(i, j)                 # Step 2: acceptance ratio

  if (u <= a) {
    return(j)                      # accept
  } else {
    return(i)                      # reject, stay at i
  }
}

# Run the Markov chain
set.seed(160)
nsim <- 101000
mcsample <- numeric(nsim)
mcsample[1] <- 0                  # Step 0: initial value X_0

for (k in 1:(nsim - 1)) {
  mcsample[k + 1] <- MH_step(mcsample[k])
}

# Burn-in
burn_in <- 1000
post <- mcsample[(burn_in + 1):nsim]  # Throw away first samples

```

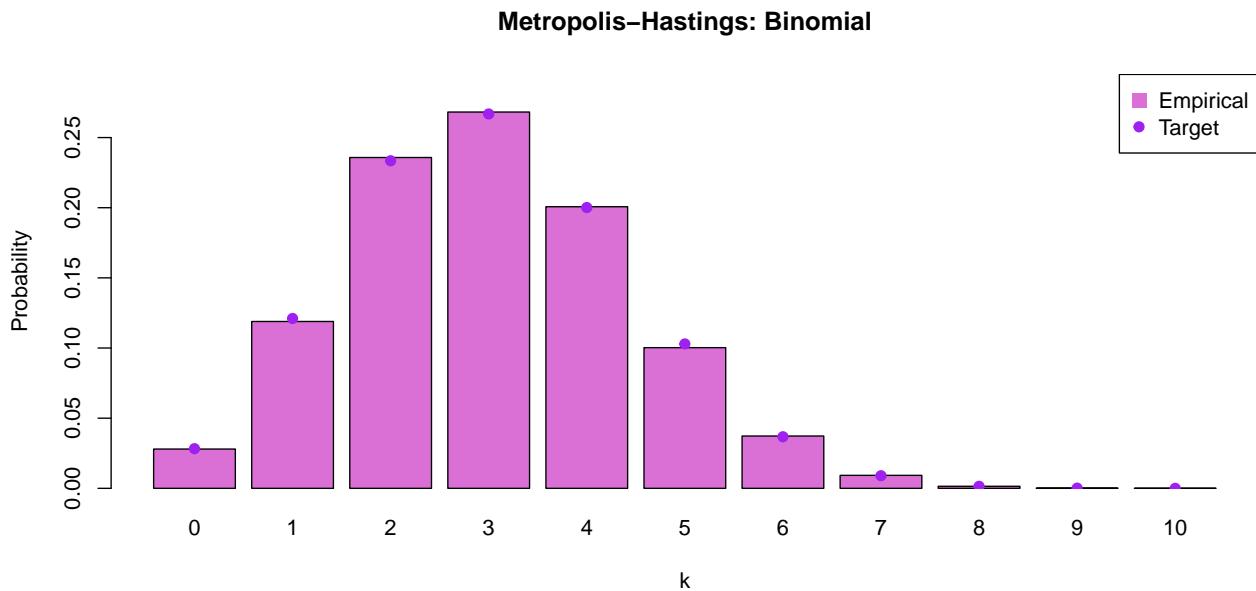
```

# Empirical distribution of the chain (ensure all states appear)
emp_counts <- table(factor(post, levels = states))
emp_freq   <- emp_counts / sum(emp_counts)

# Plot: empirical vs. target probabilities
centers <- barplot(emp_freq,
                     names.arg = states,
                     main = "Metropolis-Hastings: Binomial",
                     xlab = "k", ylab = "Probability",
                     col = "orchid",
                     ylim = c(0, max(c(emp_freq, pi_vals)) * 1.1))

points(centers, pi_vals, pch = 19, col = "purple")
legend("topright",
       legend = c("Empirical", "Target"),
       pch     = c(15, 19),
       col     = c("orchid", "purple"),
       pt.cex = c(1.5, 1))

```



```

#In case we'd like to see the results, rather than a graph
results<-t(rbind(emp_freq,pi_vals))
results

```

	emp_freq	pi_vals
0	0.02799	0.0282475249
1	0.11892	0.1210608210
2	0.23578	0.2334744405
3	0.26818	0.2668279320
4	0.20070	0.2001209490

```

5  0.10027 0.1029193452
6  0.03729 0.0367569090
7  0.00926 0.0090016920
8  0.00147 0.0014467005
9  0.00013 0.0001377810
10 0.00001 0.0000059049

```

Problem 4 - Dobrow 5.9

Show how to use the Metropolis–Hastings algorithm to simulate from the double exponential distribution, with density

$$f(x) = \frac{\lambda}{2} e^{-\lambda|x|}, \quad -\infty < x < \infty.$$

Use the normal distribution as a proposal distribution.

Solution

We want to sample from the target density

$$\pi(x) = \frac{\lambda}{2} e^{-\lambda|x|}, \quad x \in \mathbb{R}$$

We use a random-walk Gaussian proposal kernel

$$T_{x,y} = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y-x)^2}{2\sigma^2}\right),$$

which is symmetric, as we have that

$$T_{x,y} = T_{y,x}.$$

Now, we will follow the MH algorithm:

Step 0. Choose an initial state X_0 .

Step 1. Given $X_n = x$, we choose a new state y based on $T_{x,y}$

Step 2. Compute the acceptance ratio

$$a_{x,y} = \frac{\pi_y T_{y,x}}{\pi_x T_{x,y}} = \frac{\pi_y}{\pi_x} = \exp\{-\lambda(|y| - |x|)\}$$

Step 3. Let $U_{n+1} \sim U(0, 1)$ and set

$$X_{n+1} = \begin{cases} y, & \text{if } U_{n+1} \leq a(x, y) \\ x, & \text{otherwise.} \end{cases}$$

We will repeat steps 1 through 3. The stationary distribution of $\{X_n\}$ is π .

Now, we will implement the algorithm. Notice that we will discard the first few samples to eliminate the bias caused from choosing the arbitrary intial value.

```
#Parameters
lambda <- 3
sigma  <- 1

#T(x,y)
T <- function(x){
  # Draw Y ~ T(x, ·)
  y <- rnorm(1, mean = x, sd = sigma)
  return(y)
}

#a(x,y)
a_xy <- function(x, y){
  exp(-lambda * (abs(y) - abs(x)))
}

#Metropolis-Hastings step
MH_step <- function(x){
  y <- T(x)      # Step 1: propose Y
  u <- runif(1)    # Uniform(0,1)
  a <- a_xy(x, y)  # Step 2: acceptance ratio

  if(u <= a){
    return(y)          # accept
  } else{
    return(x)          # reject
  }
}

#Now we run the MC
set.seed(160)
nsim <- 51000
mcsample <- numeric(nsim)
mcsample[1] <- 0      # Step 0: initial value

for(n in 1:(nsim - 1)){
  mcsample[n + 1] <- MH_step(mcsample[n])
}

# Burn-in
burn_in <- 1000
post <- mcsample[(burn_in + 1):nsim] #Throw away first samples

#Plot
hist(post, prob = TRUE, breaks=80,
```

```

main = "Metropolis-Hastings: Double Exponential",
xlab = "x", col="lightgreen")

curve(lambda/2 * exp(-lambda * abs(x)),
      from = min(post), to = max(post),
      add = TRUE, col = "red", lwd = 2)

```

Metropolis–Hastings: Double Exponential

