

Lab 8: Changing My Shape, I Feel Like an Accident

36-350, Statistical Computing

SOLUTIONS

1. ANSWER:

```
> protein <- matrix(c(0.95,0.05,0,0.4,0.02,0.58,0,0.5,0.5),byrow=TRUE,nrow=3)
> protein
      [,1] [,2] [,3]
[1,] 0.95 0.05 0.00
[2,] 0.40 0.02 0.58
[3,] 0.00 0.50 0.50
```

2. ANSWER: The function picks a new state given the current state x and a transition matrix q . The first line checks that q is a square matrix with non-negative entries. The second line takes the number of states from the size of the matrix. The third line checks that all the rows sum to 1. The fourth line checks that x is a single whole number between 1 and the number of states. Then the new state is picked according to probabilities given by the corresponding row of the matrix.

3. ANSWER:

```
rmarkov <- function(n,q,x.initial) {
  x <- vector(length=n)
  x[1] <- x.initial
  for (t in 2:n) {
    x[t] <- chain.transition(x[t-1],q)
  }
  return(x)
}
```

4. ANSWER:

```
> seq1 <- rmarkov(n=20,q=protein,x.initial=1)
# Default histogram plot settings are ugly here; tweaks are optional
> hist(seq1,probability=TRUE,breaks=c(0,1,2,3))
> table(seq1)
```

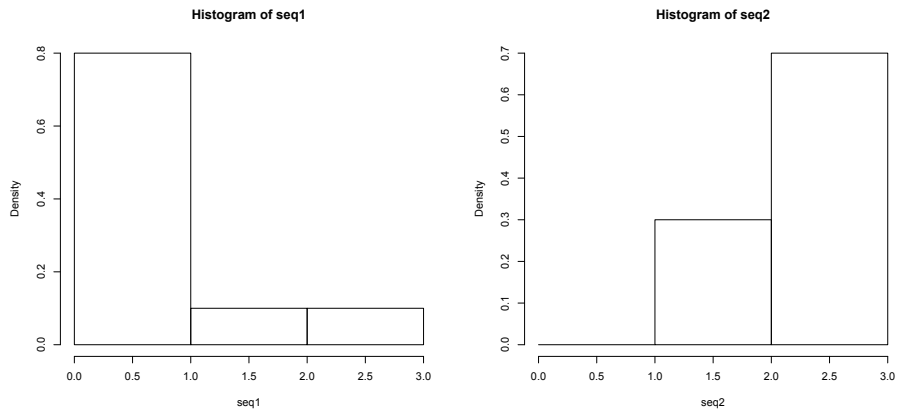


Figure 1: Left: histogram of the length-20 sequence starting from state 1; right: the same but starting from state 2.

```
seq1
 1  2  3
16  2  2
> table(seq1)/length(seq1)
seq1
 1  2  3
0.8 0.1 0.1
```

The `table()` function applied to a vector finds all the distinct values of the data and counts how often they occur. Dividing through by the length of the vector gives proportions.

5. ANSWER:

```
> seq2 <- rmarkov(n=20,q=protein,x.initial=2)
> table(seq2)/length(seq2)
seq2
 2  3
0.3 0.7
> hist(seq2,probability=TRUE,breaks=c(0,1,2,3))
```

This sequence differs because, according to the transition rates, state 1 is very persistent *if* it is reached, but it is hard to reach from state 3. This trajectory starts from state 2 and alternates between that and state 3.

6. ANSWER:

```
> seq3 <- rmarkov(n=1e5,q=protein,x.initial=1)
```

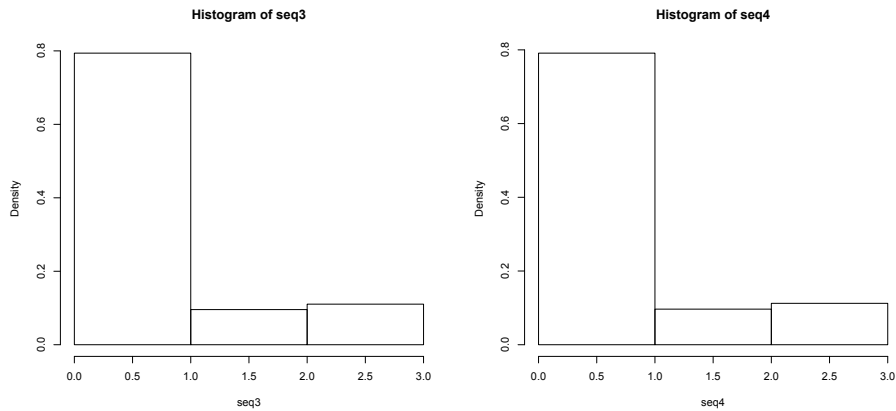


Figure 2: Histograms from sequences of length 100,000, starting from state 1 (on the left) and state 2 (on the right).

```
> seq4 <- rmarkov(n=1e5,q=protein,x.initial=2)
> table(seq3)/length(seq3)
seq3
      1      2      3
0.79386 0.09567 0.11047
> table(seq4)/length(seq4)
seq4
      1      2      3
0.79113 0.09656 0.11231
> hist(seq3,probability=TRUE,breaks=c(0,1,2,3))
> hist(seq4,probability=TRUE,breaks=c(0,1,2,3))
```

The distributions are now much more similar, differing only at the 3rd and higher significant digits. The histograms (Figure 2) are visually identical. All of this indicates that the initial state has been forgotten, and both trajectories have become representative of the long-run distribution.

7. SOLUTION: First let's look at the eigenvalues and eigenvectors, using `t()` to transpose the matrix and get the appropriate eigenvectors.

```
> eigen(t(protein))
$values
[1] 1.0000000 0.8105215 -0.3405215

$vectors
      [,1]      [,2]      [,3]
[1,] 0.9821638 -0.8043135 0.2471928
[2,] 0.1227705 0.2804611 -0.7975191
```

```
[3,] 0.1424138 0.5238524 0.5503263
```

The first eigenvector goes with eigenvalue 1, so it is the one which gives the invariant distribution. We just need to normalize it:

```
> v1 <- eigen(t(protein))$vectors[,1]
> v1/sum(v1)
[1] 0.7874016 0.0984252 0.1141732
```

This is, of course, very close to the distributions we got from the two long samples.

8. ANSWER: The distance from equilibrium after t steps is $\approx \lambda_2^t$. Setting this equal to 0.001 and solving for t gives $t = \log(0.001)/\log(\lambda_2)$.

```
> lambda2 <- eigen(t(protein))$values[2]
> log(0.001)/log(lambda2)
[1] 32.88195
```

This indicates that after a bit more than 32 steps, the chain should have mostly forgotten the initial state.