The University of British Columbia

Data Science 581 Modelling and Simulation II

Lab Assignment 4

In this lab, the TA will demonstrate exercises 1, 3, 4, and 6, not including 6(g). Please submit your answers to exercises 2, 5 and 6(g).

1. Suppose $\{\pi_i, i = 0, \pm 1, \pm 2, \ldots\}$ is a set of positive real numbers with $\sum_{i=-\infty}^{\infty} \pi_i = 1$, and set

   $$P_{i,j} = \frac{1}{3} \min\left(\frac{\pi_j}{\pi_i}, 1\right), \text{ for } j = i-1, i+1$$

   and 0 for $|j - i| > 1$. $P_{i,i}$ is obtained by subtracting the sum of all off-diagonal elements of the ith row from 1.

   Verify that the Markov chain with transition matrix P is reversible by showing that

   $$\pi_i P_{i,i+1} = \pi_{i+1} P_{i+1,i}$$

2. Suppose $\pi_i = k/(i+1)^3$ for $i > 0$ and $\pi_i = 0$ for all $i < 1$. k is a constant that ensures that $\sum_{i=1}^{\infty} \pi_i = 1$. Estimate k, using the reversible Markov chain method discussed in class. That is, the transition matrix has off-diagonal elements given by

   $$P_{i,j} = \frac{1}{6} \min\left(\frac{\pi_j}{\pi_i}, 1\right), \text{ for } j = i-2, i-1, i+1, i+2$$

   and 0 for $|j - i| > 2$. You can use the R code supplied in the lecture slides.

3. We will study the random walk Metropolis-Hastings, as programmed in the *LearnBayes* package in the context of the following problem:

   Suppose X is a normal random variable with mean 0 and unknown variance σ^2. We can estimate σ^2 as X^2. (Why?) We would also like to assess the error in such an estimate. Until now, we have used confidence intervals for this purpose. Confidence intervals have a peculiar interpretation, since they are based on probabilistic assumptions, but the interval created is not random in the sense that we can say that the probability that the true value of σ^2 lies in the interval with a given probability. Instead, we use the term "confidence" which is somewhat awkward.

   The Bayes approach avoids this awkwardness and allows for a direct probabilistic interpretation of intervals that it constructs. The tradeoff is that we have to specify our prior beliefs about a parameter beforehand, and this is often subjective, which is a matter of considerable discomfort for many analysts until they realize that much of the "science" of data analysis is based on subjective belief.

   Install the R package *LearnBayes*, and load it into an R session by typing

   ```
   library(LearnBayes)
   ```

4. Suppose you will observe a data value X, from a normal population with mean 0 and variance σ^2. If we want to make a probability statement about σ^2, we will need to view it, not as a fixed parameter, but as a quantity which we are uncertain about: a random variable with a probability distribution.

   Before we use the data, we need to specify what we believe about the distribution of X. It may be that X comes from a context which has already been studied by others, so we might be able to construct such a *prior* distribution from such information. Otherwise, we may have to come up with an arbitrary guess.

Since σ^2 is a positive quantity, we should suppose that it comes from a distribution on the positive real numbers, such as the exponential distribution:

$$f_{\sigma^2}(x) = \lambda e^{-\lambda x}, \qquad x > 0.$$

Here, λ is another parameter, which we might also model, or which we specify directly. To model this prior distribution on the log-scale, we will use the following code snippet:

```
    logprior <- dexp(sigma2, rate = lambda, log = TRUE)
```

We will be running a Markov chain for possible values of σ^2 coming from the posterior distribution - incorporating data and the prior specified above. For our example, we will suppose $X = -25$ is our observed data point.

In order to construct the log posterior distribution, we need to add the log prior to the log likelihood. The log likelihood for σ^2 when modelling from the normal population described above with the data point X can be evaluated with the following snippet:

```
    loglike <- dnorm(X, sd=sqrt(abs(sigma2)), log = TRUE)
```

Using these snippets, we code the log posterior as a function of σ^2 and the data point X and the λ parameter, into an R function as follows:

```
logNormExp <- function(sigma2, datapar) {
    X <- datapar$data
    lambda <- datapar$lambda
    loglike <- dnorm(X, sd=sqrt(abs(sigma2)), log = TRUE)
    logprior <- dexp(sigma2, rate = lambda, log = TRUE)
    return(loglike + logprior)
}
```

Supposing the observed value of X is -25 and that we want to start our Markov chain of σ^2 values with the value 300, the `datapar` object is set up as a list:

```
data <- -25
start <- matrix(300, nrow=1)
datapar <- list(data=data, lambda=1/10)
```

We also need to specify the number of Markov chain iterations, the variance-covariance (1×1) matrix and the scale:

```
m <- 10000
varcov <- matrix(1, nrow=1)
proposal <- list(var=varcov, scale = 50)
```

We now run the Metropolis-Hastings algorithm to generate the simulated σ^2 values from the posterior distribution:

```
s <- rwmetrop(logNormExp, proposal, start, m, datapar)
```

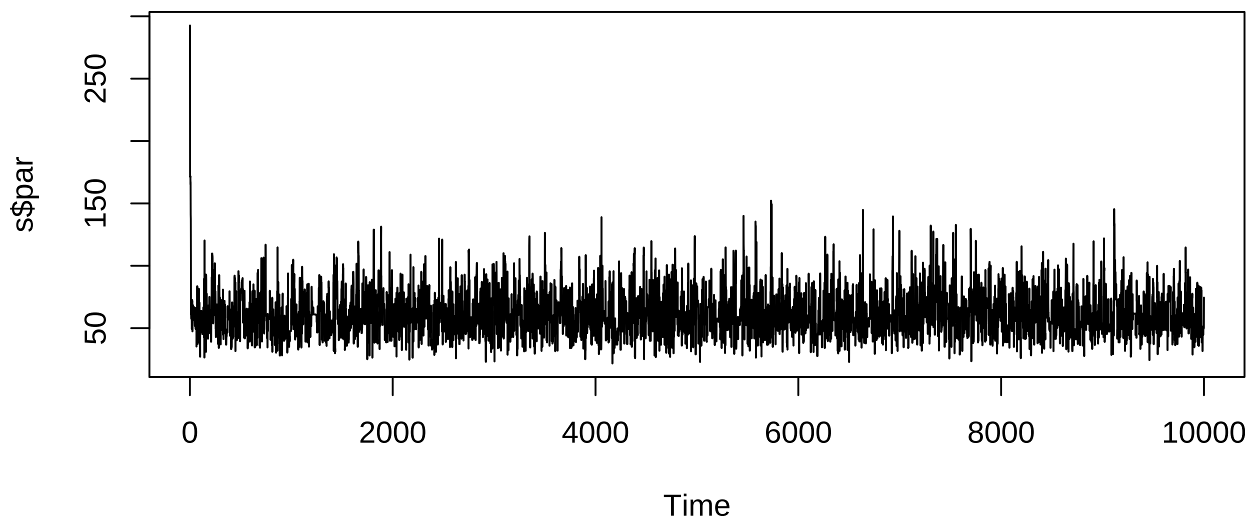
The acceptance rate should be an intermediate value:

```
s$accept

## [1] 0.3874
```

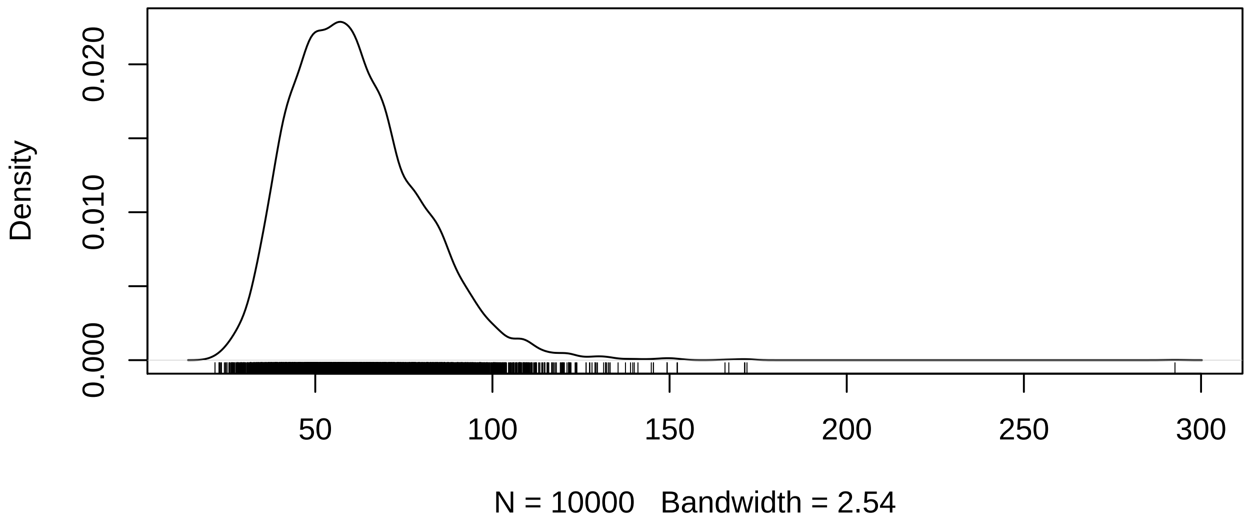
The following shows the trace of the Markov chain, showing the burn-in period and the convergence of the Markov chain to steady state:

```
par(mar=c(4, 4, 1, 1))
ts.plot(s$par)
```



Finally, we plot an estimate of the posterior density function, with the rug plot of simulated σ^2 values:

```
par(mar=c(4, 4, 1, 1))
plot(density(s$par), main = " ")
rug(s$par)
```



Note that the estimate of σ^2, based on the data point X alone is $X^2 = 625$. The mean of the prior distribution is 10, so the posterior distribution provides an intermediate type of result: the values tend to be between these two extremes.

5. Re-run the MCMC example from above, using a starting value of 625 and a λ value of $1/625$. Adjust the scale so that you get an acceptance rate of around 40%. How does the posterior distribution look now?

6. Refer to the single-player Monopoly simulation from chapter 1 of the textbook (Modeling and Simulation), and if you are not familiar with the game, consult with a classmate who is familiar. This problem concerns a greatly simplified version of the game, where a player will move a number of spaces around a 40 space rectangular board, corresponding to what appears on a pair of rolled dice. Thus, there are 40 states in the state space, with the first space corresponding to "Go". The 11th space corresponds to "Jail" and the 31st space corresponds to "Go to Jail".
   (a) Construct a 40×40 transition matrix for this Markov chain, based on the dice rolls as well as using the fact that when the player hits space 31, the probability is 1 that the player moves to space 11 at the next move. (This rule is slightly different from the usual rules, but makes analysis slightly easier with little loss of accuracy. We also assume that the player leaves space 11 at the next move, unlike the actual rules of the game.)
   (b) Find the long-run distribution for this Markov chain, calling the resulting state vector `PI`. Use `names(PI) <- 1:40` to attach names to the vector, and display the probabilities with a bar plot. Which space is second-most frequently visited? How often is space 40 visited?
   (c) Suppose you have built hotels on spaces 17, 19 and 20, and you can receive revenue of \$950, \$950 and \$1000 for each time that your opponent lands on these spaces. Calculate the long run average amount of revenue (per turn) that you could obtain from your opponent?
   (d) Suppose your opponent has built hotels on spaces 38 and 40, and it will cost you \$1500 and \$2000 each time you land on one of these spaces. Calculate the long run average cost per turn. By subtraction, you can find the expected profit per turn for someone who owns hotels on spaces 17, 19 and 20 while playing against someone with hotels on spaces 38 and 40.
   (e) Calculate the long run variance of your costs and of your revenues. Add these variances together to obtain the variance of your profit. Finally, take the square root to obtain the standard deviation of your profit. Interpret your results.

(f) Repeat the previous three exercises, under the assumption that you additionally have built hotels on spaces 22, 24 and 25 and can generate revenue \$1050, \$1050, and \$1100 at these locations, and your opponent has additionally built hotels on spaces 32, 33 and 35 which will cost you \$1275, \$1275 and \$1400 when you land on those spaces.

(g) Under the assumptions of (c) and (d), write code that will simulate concurrent Markov chains for you and your opponent, both starting in space 1, with both starting with a cash surplus of \$5000, and ending when the first person loses all of their cash. Run this simulation 1000 times, and estimate the probability that you would win. Re-run the simulation 1000 times under the assumptions in (f), and estimate the probability that you would win.