R Programming: Worksheet 8

In this final class you will produce more complicated bits of code, consolidating all the skills you have learned so far. You will translate algorithms into code without necessarily understanding all the details.

1. **Metropolis-Hastings**

   Given a (potentially multi-dimensional) probability distribution $\pi(x)$, a Metropolis-Hastings algorithm attempts to construct a Markov chain whose **stationary distribution** is π. This means if we obtain samples $X_1, X_2, \dots$ from the Markov chain, the distribution of X_k for large k should be approximately the same as π.

   To do this, we generate new samples as follows. Suppose we start at x_1, then for $i = 1, \dots, N$:

   (i) *propose* a new sample y^* using some proposal distribution $y^* \sim q(y|x_i)$;

   (ii) then compute
   $$\alpha_i = \min\left\{1,\ \frac{\pi(y^*) \cdot q(x_i|y^*)}{\pi(x_i) \cdot q(y^*|x_i)}\right\};$$
   generate a random uniform $U_i \sim U[0,1]$; if $U_i < \alpha_i$ then set $x_{i+1} = y^*$ (accept the move), otherwise set $x_{i+1} = x_i$ (reject).

   A typical choice of proposal distribution is to use a multivariate normal distribution centred on x_i.

   (a) Implement Metropolis-Hastings for a one-dimensional double exponential distribution with density
   $$\pi(x) = \tfrac{1}{2}e^{-|x|}, \qquad x \in \mathbb{R}.$$
   For the proposal use $y^* \sim N(x_i, \sigma^2)$ [does α simplify in this case?], and play around with different values of σ.

   Plot histograms and trajectories (i.e. the usual 1-D plot for numeric vectors) of the output for a few runs with σ ranging from 0.01 to 100.

   (b) Check the mean and variance of your samples for a chain of length 10,000 and $\sigma = 1$ above. Compare them to the mean and variance of the distribution π.

   (c) Now generalise your code so that it will take any univariate function π, and run the Metropolis Hastings algorithm on it, with default value $\sigma = 1$.

   Try the method with the following functions:
   $$\pi(x) = \frac{1}{1+|x|^5} \qquad\qquad \pi(x) = \frac{e^{-x^2}}{1+x^2}$$
   Use it to estimate the mean and variance of the resulting distributions. Try using the function `integrate()` to compare your answers to the true mean and variance.

   (d) Extend your functions to allow a user to specify a **burn-in** B, and a **thinning factor** K; the chain should run for $K(N+B)$ samples, saving only every Kth sample; and then discard the first B so that N samples remain. The value of B should default to $\lfloor N/10 \rfloor$, and $K = 1$.

(e) Allow your function to default to the proposal distribution above, but allow for more general specifications

(f) * Now write a function which returns an object of class `myMCMC`, which is a list containing:

- the details of the call (N, B and K);
- the chain itself (of length N);
- the proportion of times the unthinned chain 'moved' (i.e. what proportion of times was $U_i < \alpha_i$?).

Produce a suitable summary and plot method for the object.

2. **Discrete Markov Chains**

Given a finite state-space $I = \{1, 2, \ldots, k\}$, a **discrete Markov chain** is a sequence of random variables $X_0, X_1, X_2, \ldots$ taking values in I, such that

$$P(X_{t+1} = j | X_t = i) = p_{ij} \qquad i, j \in I,$$

and such that X_{t+1} is independent of $X_{t-1}, X_{t-2}, \ldots, X_0$ conditional on X_t. The $k \times k$ matrix $P = (p_{ij})$ is called the **transition matrix**.

(a) What properties should a valid transition matrix have? Write functions to (i) check whether a function is a valid transition matrix; (ii) generate a random transition matrix of a given size.

(b) Write a function to simulate observations from a discrete Markov chain, given its transition matrix. Let X_0 be sampled uniformly from the states I.

(c) Modify your answer to the previous part so that it has an argument `start`, which is either a value in I to start at, or a probability distribution to generate a starting value from. Make the default behaviour as before.

(d) Generate a large number (say 10,000) of independent chains of length 20 from some transition matrix, and look at the distribution of the states at different time points X_{20}. Try some different starting points.

(e) Compare the distribution you observe to the **normalised** eigenvector of $I_d - P^T$ corresponding to eigenvalue 0 (here I_d is the identity matrix). Why is there always an eigenvalue of 0?

(f) Try with the following transition matrices and comment:

$$P_1 = \begin{pmatrix} 0.01 & 0.98 & 0.01 \\ 0.01 & 0.01 & 0.98 \\ 0.98 & 0.01 & 0.01 \end{pmatrix} \qquad P_2 = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$