STA 4503, Spring 2013 — Programming Assignment #1

Preferably handed in in class on March 8, but you can hand it in instead during office hours March 11 (3-4pm) if you like. Please hand it in on 8 1/2 by 11 inch paper, stapled in the upper left, with no other packaging.

This assignment is to be done by each student individually. You may discuss it in general terms with other students, but the work you hand in should be your own. In particular, you should not leave any discussion with someone else with any written notes (either on paper or in electronic form).

For this assignment, you will write a program to sample from the posterior distribution of the Bayesian model below, using single-variable Metropolis updates, and try it out on three datasets that are provided on the course web page, http://utstat.utoronto.ca/∼radford/sta4503/

The model is a simple linear regression model with two covariates, defined as follows:

$$\begin{aligned}
\log \sigma_e &\sim N(0, 2^2) \\
\log \sigma_\beta &\sim N(0, 2^2) \\
\beta_0 &\sim N(0, 10^2) \\
\beta_1 | \sigma_\beta &\sim N(0, \sigma_\beta^2) \\
\beta_2 | \sigma_\beta &\sim N(0, \sigma_\beta^2) \\
y_i | x_{i1}, x_{i2}, \sigma_e, \beta_0, \beta_1, \beta_2 &\sim N\left(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2},\, \sigma_e^2\right), \quad \text{for } i = 1, \ldots, n
\end{aligned}$$

In the above specification, variables are assumed independent except for the dependencies shown, and those that they imply.

You should write a program to simulate a Markov chain based on single-variable Metropolis updates, applied to each of the five parameters in turn, that converges to the posterior distribution of $(\log \sigma_e,\ \log \sigma_\beta,\ \beta_0,\ \beta_1,\ \beta_2)$ given a dataset $(y_1, x_{11}, x_{12}), \ldots, (y_n, x_{n1}, x_{n2})$. The proposal distribution for changing one of the components of the state should be normal with mean equal to the current value of that component and a standard deviation that you should adjust to get a reasonably large (but not too large) acceptance rate (somewhere between 0.2 and 0.6 is probably about right). Your program should allow for different proposal standard deviations for the updates to different components. Note that the first two updates should be for $\log \sigma_e$ and $\log \sigma_\beta$, not for σ_e and σ_β. Your program should keept track of how many proposals are accepted for changes to each component, and output the acceptance rates for the five components at the end, so that you can adjust the proposal standard deviations as necessary. Your program should return a matrix with five columns for the five components, with each row the value after each iteration.

For each of the three datasets, you should do one run from an initial state of all zeros, at least three runs from states set randomly from independent $N(0, 1)$ distributions, and runs from any other initial states you think would be good to try. You should explicitly set the random number seed for each run so that you can reproduce your results. Your runs should be for at least 10,000 iterations (where each iteration consists of an update of all five components of the state), longer if that seems desirable. You will probably need to do several preliminary runs in order to choose appropriate proposal standard deviations, and to see how many iterations are necessary in order to be confident that the chain has converged to the correct distribution. This will require looking at the trace plots described next.

For each dataset, you should look at a trace plot of each of the five components — that is, at a plot of that component's value versus iteration number. Trace plots for other functions of state (such as the log likelihood) may also be of interest. You should decide from these plots how many iterations it would be appropriate to discard as "burn-in" iterations, for which approximate convergence has not been reached. You should then look at pairwise scatterplots of the parameter values in iterations after the burn-in period, in order to see what the posterior distribution looks like. For all these plots, you may decide to "thin" the output (eg, take only every tenth iteration) if otherwise the number of points would be unmanageably large.

You should discuss the results with these three datasets, handing in suitable plots to illustrate your conclusions. Among the questions to discuss are: Do runs from different initial states seem to converge to the same distribution in a reasonable number of iterations, or would more iterations than you can feasibly do be required? If convergence is reached in a reasonable time, how many burn-in iterations need to be discarded? What are the characteristics of the posterior distributions for these datasets, and are these characteristics related to the convergence of the Markov chains?

You should also hand in your program and test scripts, which may be in any language, as long as you're not using any built-in MCMC facilities.