Penn State Astrostatistics MCMC tutorial

Murali Haran, Penn State Dept. of Statistics

Bayesian change point model: full conditional distributions

Our goal is to draw samples from the 3-dimensional **posterior** distribution $f(k, \theta, \lambda \mid \mathbf{Y})$ The posterior distribution is

$$\begin{aligned} f(k, \theta, \lambda \mid \mathbf{Y}) \propto \prod_{i=1}^{k} \frac{\theta^{Y_i} e^{-\theta}}{Y_i!} \prod_{i=k+1}^{n} \frac{\lambda^{Y_i} e^{-\lambda}}{Y_i!} \frac{1}{n-2} 1(k \in \{2, 3, \ldots, n-2\}) \\ \times \frac{1}{\Gamma(0.5) b_1^{0.5}} \theta^{-0.5} e^{-\theta/b_1} \times \frac{1}{\Gamma(0.5) b_2^{0.5}} \lambda^{-0.5} e^{-\lambda/b_2} \end{aligned} \tag{1}$$

$1(k \in \{2, 3, \ldots, n-2\})$ is an indicator function, meaning it is 1 if $k \in \{2, 3, \ldots, n-2\}$ and 0 otherwise.
Note: The reason we have a formula for what f is proportional to (hence $\propto$ rather than $=$) instead of an exact description of the function is because the missing constant (the normalizing constant) can only be computed by integrating the above function. Fortunately, the Metropolis-Hastings algorithm does not require knowledge of this normalizing constant.

From (1) we can obtain full conditional distributions for each parameter by ignoring all terms that are constant with respect to the parameter. Sometimes these full conditional distributions are well known distributions such as the Gamma or Normal. That occurs when using "conjugate priors" and can simplify the construction of the MCMC algorithm though it does not necessarily lead to a more efficient algorithm.
Full conditional for θ:

$$\begin{aligned} f(\theta \mid k, \lambda, \mathbf{Y}) &\propto \prod_{i=1}^{k} \frac{\theta^{Y_i} e^{-\theta}}{Y_i!} \times \frac{1}{\Gamma(0.5) b_1^{0.5}} \theta^{-0.5} e^{-\theta/b_1} \\ &\propto \theta^{\sum_{i=1}^{k} Y_i - 0.5} e^{-\theta(k+1/b_1)} \end{aligned}$$

by staring at the form of the density $\propto \text{Gamma}\left(\sum_{i=1}^{k} Y_i + 0.5, \frac{b_1}{kb_1 + 1} \right)$

Full conditional for λ:

$$f(\lambda \mid k, \theta, \mathbf{Y}) \propto \prod_{i=k+1}^{n} \frac{\lambda^{Y_i} e^{-\lambda}}{Y_i!} \times \frac{1}{\Gamma(0.5) b_2^{0.5}} \lambda^{-0.5} e^{-\lambda/b_2}$$

by staring at the form of the density $\propto \text{Gamma}\left(\sum_{i=k+1}^{n} Y_i + 0.5, \frac{b_2}{(n-k)b_2 + 1}\right)$

Full conditional for k:

$$\begin{aligned} f(k \mid \theta, \lambda, \mathbf{Y}) &\propto \prod_{i=1}^{k} \frac{\theta^{Y_i} e^{-\theta}}{Y_i!} \prod_{i=k+1}^{n} \frac{\lambda^{Y_i} e^{-\lambda}}{Y_i!} 1(k \in \{2, 3, \dots, n-2\}) \\ &\propto \theta^{\sum_{i=1}^{k} Y_i} \lambda^{\sum_{i=k+1}^{n} Y_i} e^{-k\theta - (n-k)\lambda}. \end{aligned}$$

We are now in a position to run the Metropolis-Hastings algorithm.

Note: θ, λ have full conditional distributions that are well known and easy to sample from. We can therefore perform Gibbs updates on them where the draw is from their full conditional. However, the full conditional for k is not a standard distribution so we need to use the more general Metropolis-Hastings update instead of a Gibbs update.

The Metropolis-Hastings algorithm:

1. Pick a starting value for the Markov chain, say $(\theta^0, \lambda^0, k^0) = (1, 1, 20)$.
2. 'Update' each variable in turn:
   (a) Sample $\theta^i \sim f(\theta \mid k, \lambda, \mathbf{Y})$ using the most up to date values of k, λ (Gibbs update using the derived Gamma density).
   (b) Sample $\lambda^i \sim f(\lambda \mid k, \theta, \mathbf{Y})$ using the most up to date values of k, θ. (Gibbs update using the derived Gamma density).
   (c) Sample $k \sim f(k \mid \theta, \lambda, \mathbf{Y})$ using the most upto date values of k, θ, λ. This requires a Metropolis-Hastings update:
      i. 'Propose' a new value for k, k^* according to a proposal distribution say $q(k \mid \theta, \lambda, \mathbf{Y})$. In our simple example we pick $q(k \mid \theta, \lambda, \mathbf{Y}) = \text{Unif}\{2, \ldots, m-1\}$ where m is the length of the vector (time series) $\mathbf{Y}$.
      ii. Compute the Metropolis-Hastings accept-reject ratio,
      $$\alpha(k, k^*) = \min\left(\frac{f(k^* \mid \theta, \lambda, \mathbf{Y})q(k \mid \theta, \lambda, \mathbf{Y})}{f(k \mid \theta, \lambda, \mathbf{Y})q(k^* \mid \theta, \lambda, \mathbf{Y})}, 1\right)$$
      iii. Accept the new value k^* with probability $\alpha(k, k^*)$, otherwise 'reject' k^*, i.e., the next value of k remains the same as before.
   (d) You now have a new Markov chain state $(\theta^1, \lambda^1, k^1)$
3. Return to step #2 N-1 times to produce a Markov chain of length N.

Now if we want to produce an approximation to any expectation with respect to the posterior, say $E_f(g(X))$. Simple examples of expectations for the above problems are $E_f(\theta \mid \boldsymbol{Y})$, $E_f(\lambda \mid \boldsymbol{Y})$, $E_f(k \mid \boldsymbol{Y})$ or slightly more complicated ones like $P_f(\theta < 5 \mid \boldsymbol{Y})$, or $E_f(\lambda/\theta \mid \boldsymbol{Y})$. Using MCMC, these are easy to approximate by just taking averages of the samples generated by the algorithm. It is also easy to obtain an approximation to their distribution, for example to look at the posterior distribution of $\lambda/\theta \mid \boldsymbol{Y})$, one can just look at a histogram of the ratio of λ and θ values produced by the algorithm.

That's essentially all there is to MCMC recipe basics! Important details to work out:

- How do we choose N so we feel confident in our approximations based on the MCMC algorithm?

- How do we try to ensure that our starting values have not had an undue influence on our results?

Both the above questions require getting a sense of how reliable our approximations are. Like in most of statistics, this involves getting a handle on the bias and the variance of the Monte Carlo approximation.