Metropolis Hastings Algorithm and Slice Sampling

Chung Him Kwok

2023-03-06

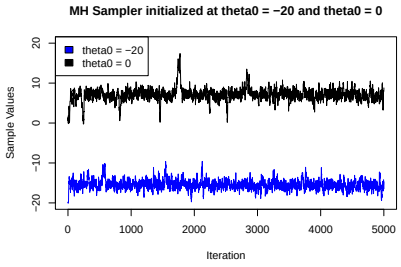
Introduction

In this project, we implement a Markov Chain Monte Carlo (MCMC) algorithm called Slice Sampling to sample from an intractable un-normalized posterior density. In particular, the posterior density we sample from is given by $\pi(\theta|y) = \Pi_{i=1}^{n} \frac{1}{1+(y_i-\theta)^2}$ with $n = 4$ and $y = (y_1, y_2, y_3, y_4) = (-16.6, -14.7, 6.3, 8.4)$, which corresponds to posterior of a Cauchy likelihood with a uniform prior.

We observed that this posterior is heavy-tailed and multi-modal, which makes it non-trivial to sample from. We first adopt a Metropolis-Hasting (MH) algorithm, followed by a Slice Sampling (with Rejection Sampling), and compare their performances. Finally, we wrap up the project by making approximate inference on the posterior with the better sampler.

Metropolis Hastings (MH) Algorithm

We first implement two Random Walk MH samplers, with $\theta_0 = -20$ and $\theta_0 = 0$ respectively, as our baseline. In particular, we use the following proposal distribution for each iteration $q(\cdot|\theta) = \mathcal{N}(\theta, 1)$.



We observed that, since the objective distribution is multi-modal, our MH sampler has a high tendency to get stuck in one of the modes and fail to explore over the other parts of the function. Our implementation shows that different initializations of θ_0 got us stuck in different modes and it might take a time time for the sampler to escape from one mode to another, leading to an extremely long burn-in period and poor mixing.

Slice Sampling

Therefore we suggest and implement an alternative MCMC algorithm, Slice Sampling, invented by Neal (2003), which does not require an explicit proposal distribution. To implement a Slice Sampler, we first need to implement a Rejection Sampler which we will be using to sample from the super-level-set. We then run our full Slice Sampling algorithm with the same two initializations as above and make observations.

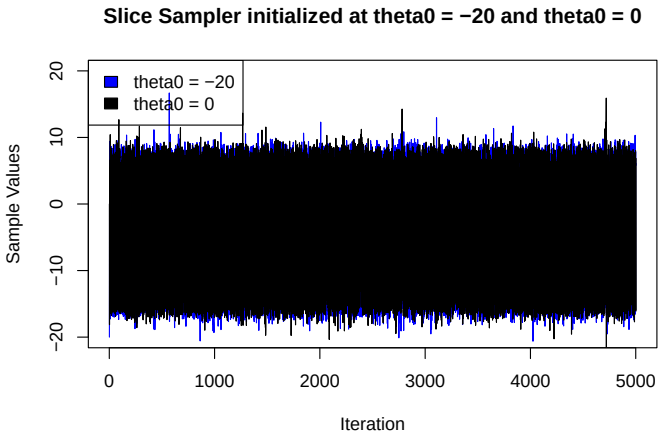
Rejection Sampling

The trick to sampling from the super-level-set $G_s = \{\pi(\theta) \geq s\}$ is by noting that two sufficient but not necessary conditions for $\pi(\theta)) < s$ is given by 1) $(\theta < y_1 = -16.4) \wedge (\pi(\theta) < (\frac{1}{1+(-16.4-\theta)^2})^4)$, or 2) $(\theta > y_4 = 8.4) \wedge (\pi(\theta) < (\frac{1}{1+(8.4-\theta)^2})^4)$, which corresponds to the interval given by $(y_1 - (s^{-1/4} - 1)^{1/2}, y_4 + (s^{-1/4} - 1)^{1/2})$ when $s^{-1/4} - 1 > 0$ (which should always be the case when $0 < s < 1$). (Note that s is uniformly drawn between 0 and π, which is always less than 1.

In a nutshell, we construct the set $H_s = (y_1 - a, y_4 + a) \supset G_s$, where $a = \sqrt{\frac{1-\alpha}{\alpha}}$ with $\alpha = s^{1/4}$ when $s^{-1/4} > 1$, and $a = 0$ otherwise. We then perform the rejection sampling on G_s using H_s. We redraw the s sample in case we obtain $s = 0$ from the uniform sampling (which happens with probability $= 0$).

Performance of the Slice Sampler

We implement the full Slice Sampler with our Rejection Sampling algorithm, with initializations $\theta_0 = -20$ and $\theta_0 = 0$, and plot the sampling results below.



Comments on Slice Sampling

Our figure depicts that the mixing our the Slice Sampling is much better than the one we had for the MH algorithm. The sampler efficiently explored around the two nodes of our posterior distribution and did not suffer from the problem of a long burn-in period. Moreover, unlike the MH sampler, which tends to get stuck in one of the nodes depending on the how we initialize θ_0, the mixing of our slice sampler is not sensitive to different initializations, as we observed that the two samplers with different initializations are yielding similar results.

In conclusion, we believe that Slice Sampling is much more efficient in sampling from our multi-modal, heavy-tail unnormalized posterior stated above.

Inference with Slice Sampler

We wrap up the project by making approximate inferences with our slice sampler. In particular, we wish to approximate two quantities given by: 1) $P(\theta > 8|y)$, and 2) $P(\theta < -15|y)$. We run the Slice Sampler with $\theta_0 = -20$ (actually we can run any of them as we showed that the initialization does not really matter for Slice Sampling) and make the estimation.

From the output of the program, we found that the approximate probabilities are given by $P(\theta > 8|y) \approx 0.0966$ and $P(\theta < -15|y) \approx 0.344$, which are consistent to our expected solutions.