Parallel Slice Sampling

Teresa Pietrabissa[1], Simone Rusconi[1]

Bayesian Young Statisticians Meeting (BAYSM), Milan June, 5-6, 2013
Paper no. 32

[1] Politecnico di Milano, Department of Mathematics, Milan, Italy
teresa.pietrabissa@mail.polimi.it
simone1.rusconi@mail.polimi.it

Abstract

To draw a sample from a probability distribution using a Markov Chain Monte Carlo (MCMC) method, there is an easy algorithm named Slice sampling. It is based on the fundamental theorem of random number generation: to get a sample X from a function g, which is a density of a finite measure (not necessarily a probability measure) over a finite-dimensional Euclidean space, it is equivalent to simulate from the marginal density of a joint uniform (U, X) under the plot of $g(x)$. The sample X is obtained by using a Gibbs-Sampler with the full conditionals: $U|X \sim \mathcal{U}(0; g(X)); X|U \sim \mathcal{U}(S), where\ S = \{x : g(x) > u\}$.

Our aim is to create a library, using the Slice sampling, to draw a MCMC sample from any density g, in particular to get a realization from the posterior density of a Bayesian model. The main problems of this algorithm are basically two: the solution of the inequality $g(x) > u$, which can be hard to find due to the irregularities the measure density g can be affected by, and the high dimensionality of the support of the density itself. We will present and discuss a solution and some statistical test applications, using a parallel language, which is, nowadays, becoming more and more commonly employed in the context of Bayesian statistical models.

Keywords: Parallel; Slice Sampling; Posterior Density.

1 Introduction

MCMC methods provide a general approach for approximating integrals with respect to a wide range of complex distributions.

In a typical Bayesian analysis, the integration measure is the posterior distribution. Let $\underline{\beta} \in \mathbb{R}^k$ be the argument of the posterior, i.e. the variable of integration. The posterior distribution usually has a density with respect to the Lebesgue measure (for instance) and is proportional to a measure density $g(\underline{\beta}) = L(\underline{\beta}|X)\pi(\underline{\beta})$, where $L(\underline{\beta}|X)$ is the likelihood, seen as function of $\underline{\beta}$ and evaluated in the data X, and $\pi(\underline{\beta})$ is the prior density of the model. Our purpose is to create a library that takes as input a generic $g(\underline{\beta})$ and returns a MCMC sample from the posterior density $\underline{\beta}^{(i)}, i = 1 \div G$.

2 The algorithm

The algorithm that we are going to develop can be summarized as follow. After initializing $\underline{\beta}^{(0)}$ as a suitable value:

1. Sample $u^{(i)}$ from $\mathcal{U}(0; g(\underline{\beta}^{(i)}))$.
2. Sample $\underline{\beta}^{(i+1)}$ from $\mathcal{U}(S^{(i)})$, where $S^{(i)} := \{\underline{\beta} \in \mathbb{R}^k | g(\underline{\beta}) > u^{(i)}\}$.

Repeat steps 1. and 2. for $i = 0 \div G - 1$ to obtain the sample desired.

MCMC methods are sequential methods as their structure requires it. The same is for the Slice Sampling where, as briefly shown, we get a new realization from the previous one. Indeed, we are sampling a Markov Chain, which is time dependent by definition. However, each evaluation of the new realization can be split into simplier steps which are parallelizable. More details on the two main steps are given as follow.

2.1 First step

The first step requires to sample from $\mathcal{U}(0; g(\underline{\beta}^{(i)}))$. This is the easiest step of the algorithm as it consists in sampling from a probability distribution which is simple to generate from. Moreover, this is a one-dimensional random number generation and, for this reason, it becomes easier to parallelize it with libraries that fully support one-dimensional random number generation itself.

2.2 Second step

The second step consists in sampling $\underline{\beta}^{(i+1)}$ from $\mathcal{U}(S^{(i)})$, which requires resolution of inequality $g(\underline{\beta}) > u^{(i)}$. Based on the computing power of GPU programming, we follow these simplier steps:

1. In a parallel way, sample $\underline{\beta}_{(j)}^{(i)}$from $\mathcal{U}(R)$, where $j = 1 \div K$, K is as big as needed, i is the main step index, $R \in \mathbb{D}$ is an appropriated support for the algorithm and $\mathbb{D} \subset \mathbb{R}^k$ is the domain of the function $g(\underline{\beta})$.
2. The new realization $\underline{\beta}^{(i+1)}$ is equal to one of the $\underline{\beta}_{(j)}^{(i)}$ that solves the inequality $g(\underline{\beta}_{(j)}^{(i)}) > u^{(i)}$.

The appropriate support R can be obtained in different ways and in general we will ask the user to provide it. Once we have defined R we can sample from k one-dimensional uniforms, one for each dimension. This way we have splitted the high dimensionality problem into simplier problems.
Finally, as applications, we will compute posterior distributions in some "benchmark" Bayesian models using Parallel slice sampling.

3 A Simple Example

We present here a simple example of the efficency of the Slice Sampling method. To do so we tried to get an MCMC sample from a beta-binomial distribution, as we already know that the prior density $\pi(\beta) = Beta(a, b)$ is conjugate with respect to the binomial likelihood, generating the posterior density $\pi(\beta \mid \underline{X}) = Beta(a + \sum_{i=0}^{n} X_i, b + n - \sum_{i=0}^{n} X_i)$.

Using R software we drew a sample from the Likelihood: $\underline{X} \mid \beta \sim Bi(n, \beta)$, where $n = 200$ and $\beta = 0.3$; the prior density for the model, as already introduced, is the Beta distribution: $\beta \sim \pi(\beta) = Beta(a, b)$, where $a = b = 1.2$ so that our prior is uninformative with rispect to the parameter β. Using our C++ code we got the posterior sample of the model and obtained what was expected from theoretical analysis.

As shown in the traceplot, the posterior sample has a very good behaviour and it seems not to have any particular shape. If we compare the theoretical distribution with the histogram of the posterior sample, we find a very good match which states the goodness of the algorithm.

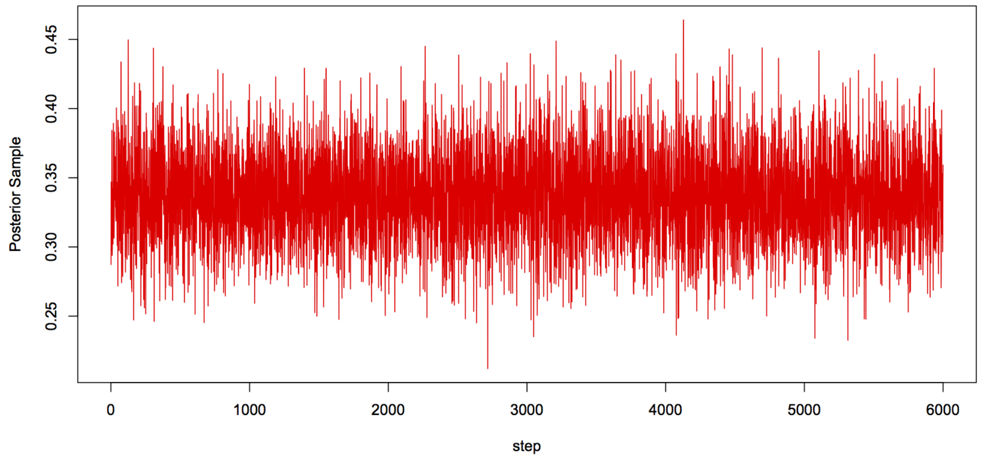

Figure 1: Traceplot for the posterior sample, obtained from the Beta-Binomial model with burning=50, thinning=3.

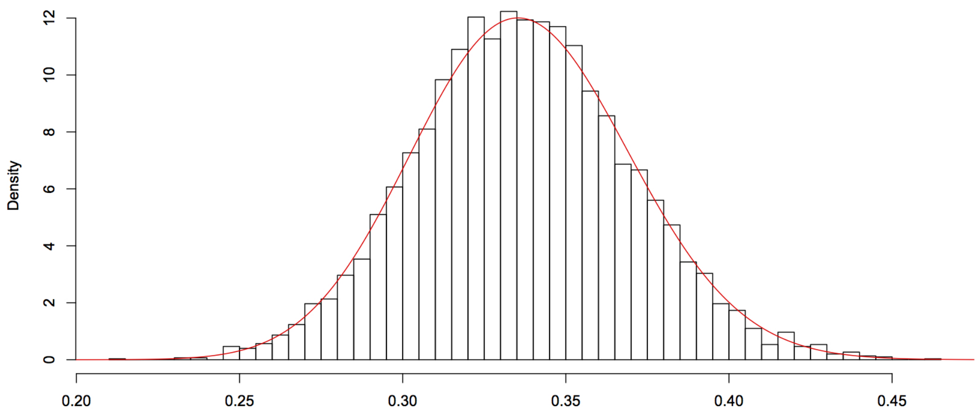

Figure 2: Comparison in between thoeretical distribution (in red) and the histogram of the posterior sample.