A Lightweight Guide on Gibbs Sampling and JAGS

Gianpaolo Coro

Istituto di Scienza e Tecnologie dell'Informazione A. Faedo, Pisa, Italy

1 Summary

In this document we give some insight about how Gibbs Sampling works and how the JAGS modelling framework implements it. The hope is that, once the reader will have understood these concepts, building a model to perform Bayesian Inference with JAGS should be much easier or, at least, the reader should be more aware of what happens *behind the scenes*. We assume the reader to have basic knowledge about probability, sufficient to understand the difference between a probability and a probability density.

This paper is organized as follows: Section 2, presents the basic principles of Bayesian Inference. Section 3, introduces the JAGS software and its basic modelling principles. Section 3.1, introduces the probabilistic graphical models on which Gibbs Sampling and JAGS are based. Section 4 reports basic definitions about Markov Chains. Section 5 explains Gibbs sampling on the basis of the previously introduced concepts. It gives the rationale behind the method and demonstrates its correctness. Furthermore, it shows how to apply Monte Carlo Integration to the output of Gibbs sampling. Section 6 reports about how JAGS is related to Gibbs sampling. Furthermore, it shows an example written with the R programming language that uses JAGS. The example demonstrates the effectiveness of the complete modelling process. Finally, Section 7 draws the conclusions.

2 Bayesian Inference

In order to define Bayesian Inference we first clarify the concept of *posterior probability distribution* (or *posterior probability density*). The posterior probability distribution of a random event is defined as the *conditional* probability that is assigned to the event after relevant evidence has been taken into account. In other words, if there is experimental evidence of a certain probabilistic phenomenon, the posterior probability distribution relates such evidence with some random variables. For example, in the case of a random variable x having a Gaussian distribution $Norm(\mu, \sigma)$ (with μ and σ being mean and standard deviation), the posterior probability distribution would be indicated as $p(\mu, \sigma|x)$.

Such distribution allows to calculate the probability of a pair (μ, σ) given a certain value of x. The analytical form of a posterior probability can be difficult to obtain, and one of the scopes of this document is to show a method to simulate its values.

Generally speaking, given a set of experimental evidences $\bar{y} = \{y_1, y_2, .., y_n\}$ linked to a set of random variables $\bar{\theta} = \{\theta_1, \theta_2, .., \theta_m\}$ by means of a probabilistic relation, the posterior probability density is indicated as $p(\bar{\theta}|\bar{y})$. Another important concept is the *likelihood* to the data, a dual concept with respect to the posterior probability distribution. Likelihood is defined as the probability density associated to the evidence $\bar{y}$ given the variables $\bar{\theta}$ and is indicated as $p(\bar{y}|\bar{\theta})$.

The two quantities are indeed related by the Bayes' rule

$$p(\bar{\theta}|\bar{y}) = \frac{p(\bar{y}|\bar{\theta})p(\bar{\theta})}{p(\bar{y})}$$

where $p(\bar{y})$ is the probability density associated to the sample data (*marginal density* of sample data or *marginal likelihood*) without considering the random variables. The term $p(\bar{\theta})$ is called the *prior* probability and indicates an *a priori* estimate of the distribution of the parameters $\bar{\theta}$ [4].

In this paper we will use P to denote probabilities and p for probability densities. Bayes' rule for probability densities is conceptually different from Bayes'rule for probabilities. In the case of probabilities, the rule is

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

where A and B are two probabilistic phenomena. The rule for distributions is derived from the definition of joint probability density and not from the axioms of probability, as it happens in the case of probabilities.

From the above considerations, we can define **Bayesian Inference as a process, based on Bayes' rule for probabilities distributions, that estimates the posterior probability density by using prior knowledge about the parameters (priors) and updates the posterior probability density estimate by means of likelihoods as long as new evidence (real data) is acquired.**

Thus, Bayesian Inference factorizes prior knowledge about the parameters and likelihoods, using likelihoods as reference to relate the parameters to the real data. Alternative methods address the simulation of the posterior probability density directly or of the likelihood only [24, 10]. Usually, Bayesian Inference is used to find the best parameters given the observed data, i.e. the parameters that maximize the posterior probability distribution. The best choice for these depend on the form of the distribution. Usually, parameters belonging to the central tendency of the posterior probability density are taken, other times the maximum is searched for (Maximum a Posteriori or MAP). The choice of the 'best' parameters is part of a more general discussion about the search for estimates that don't involve outliers [11].

When the best estimate for the parameters has been found, it can be used in many ways, for example to simulate a function (e.g. the best estimate of the coefficients of a linear combination) or to make predictions on the distribution of a new data point by means of the posterior probability density [1].

3 JAGS

JAGS (Just another Gibbs sampler) is a program developed by Martyn Plummer [20, 21] that implements Bayesian inference using Markov Chain Monte Carlo models (MCMC)[1]. It is based on a *hierarchical* models, a special case of *graphical* models, which will be described later in this Section. JAGS relies on a dialect of the BUGS programming language to define models. BUGS is a *declarative* language where the programmer is requested only to specify the known relationships among the variables. Thus, in the BUGS modelling language the order in which the relations are specified is not important, they must be only declared. The approach is different from imperative languages (e.g. R, C, Fortran, Pascal etc.), in which an algorithm is built as a sequence of steps from the computer point of view. The relations among the variables are specified in terms of probabilistic or deterministic functions. Furthermore, likelihoods are defined when real data must be taken into account. JAGS is responsible for analysing the distributions definitions and applying the most appropriate strategy to simulate the posterior probability density values.

The general approach is the one of Gibbs Sampling, described in the Section 4, but it is accompanied by several techniques that enforce (or in some cases substitute) the standard Gibbs Sampling. For example, the Metropolis–Hastings algorithm [5], the Slice sampling [17] and the Adaptive Rejection sampling [8] algorithms are included in JAGS. In Section 5 we explain how these may intervene during the process. JAGS is an alternative to other wrappers for BUGS modelling dialects, like WINBUGS [7, 19] and OpenBUGS [23]. The main aim of JAGS is to provide a multi-platform open-source framework that is easily expandable with new algorithms and is also open to criticize graphical models. JAGS is endowed with a wrapper for the R language (rjags) [21] that is able to run the BUGS model interpreter as an external program and to retrieve the output as an R object, on which further processing can be applied [9].

One example of a BUGS model is the following, which represents a simple linear regression

```
mu <- alpha + beta(x - x.bar);
Y ~ dnorm(mu, tau);
x.bar <- mean(x);
alpha ~ dnorm(0.0, 1.0E-4);
beta ~ dnorm(0.0, 1.0E-4);
tau ~ dgamma(1.0E-3, 1.0E-3);
```

The code declares the dependencies among the variables in terms either of stochastic or deterministic relations. As said before, unlike imperative languages the order of the declarations is not important. They are treated by JAGS as they were the hypotheses of a theorem to be demonstrated. JAGS autonomously ensures that the model is coherent. The above R code means that the variable *mu* is a linear combination of other variables. *Y* is a stochastic variable distributed like a Gaussian function, which depends also on other two variables, *mu* and *tau*. Also *alpha* and *beta* are distributed like a Gaussian, while *tau*

[1] MCMC is a generic term indicating an algorithm that samples from probability distributions by constructing a Markov chain that has the desired probability density as its equilibrium (or ergodic) distribution.

follows a Gamma distribution. In mathematical terms, the second relation in the model means that the following distribution must be taken into account

$$G(Y, mu, tau) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(Y-mu)^2 tau}{2}}$$

(note that for JAGS $tau = \frac{1}{\sigma^2}$).

JAGS is initialized by means of parameters which accompany the BUGS model and that may include observed data. If a variable contains real data, JAGS interprets its corresponding probability distribution as a likelihood. Referring to the example, if Y contains real observations then JAGS interprets $Y \sim dnorm(mu, tau)$ as a likelihood. The other distributions are treated either as simple *priors* or as *conditional* probability distributions. The deterministic functions act as definitions to simplify the syntax.

3.1 Graphical models

Graphical models are the basic concept under JAGS. The fundamental object in a graphical model is a node, which represents a variable in the model (either observed or unobserved). Nodes have children and parents, and a dimension attribute [2]. JAGS allows to define nodes that represent stochastic parameters (*Stochastic* nodes), deterministic parameters (*Logical* nodes) and constants (*Constant* nodes). The relations among the nodes are automatically traced on the basis of the names of the variables. Parameters depending on other parameters are seen as having these as 'parents'. Furthermore, JAGS allows to define *Array* nodes, representing containers of parameters entirely related to other variables and *Subset* nodes, related to other nodes by subscripting. Generally speaking, a graphical model is a probabilistic model in which a graph defines the conditional dependencies among the random variables. The graph gives a compact representation from which the independent and the conditioned variables are immediately evident. One example is in figure 1, where a graph represents the relations between 4 variables $\{\theta_1, \theta_2, \theta_3, \theta_4\}$. In this case, the

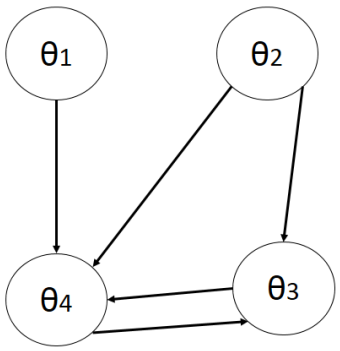

Figure 1: A probabilistic graphical model for 4 variables.

joint probability distribution for the variables is

$$p(\theta_1, \theta_2, \theta_3, \theta_4) = p(\theta_1)p(\theta_2)p(\theta_4|\theta_1, \theta_2)p(\theta_3|\theta_2, \theta_4)$$

This formula multiplies all the conditional distributions. As yet said, JAGS knows which ones must be interpreted as likelihoods and which as priors or conditional distributions. In Section 5 we show that Gibbs Sampling just starts from this joint distribution to sample the posterior probability density. By abstracting the graphical model in the example, we can define the joint distribution associated to a graphical model representing the variables $\{\theta_1, \theta_2, \ldots, \theta_n\}$, as

$$p(\theta_1, \theta_2, \ldots, \theta_n) = \prod_{i=1}^{n} p(\theta_i | par_i)$$

where par_i is the set of parents of the node θ_i. From the above formula the factorized representation coming from the usage of the graph is evident. *Hierarchical* models are a special case of graphical models, for which the edges have a *causal interpretation*, established by the hierarchical dependencies among the nodes [6]. In hierarchical models, conditional independence relations are imposed by the hierarchy. JAGS models are indeed hierarchical models.

Figure 2 reports the hierarchical model of the linear regression reported at the beginning of this Section, where the squares indicate deterministic functions, while the circles indicate probabilistic distributions.

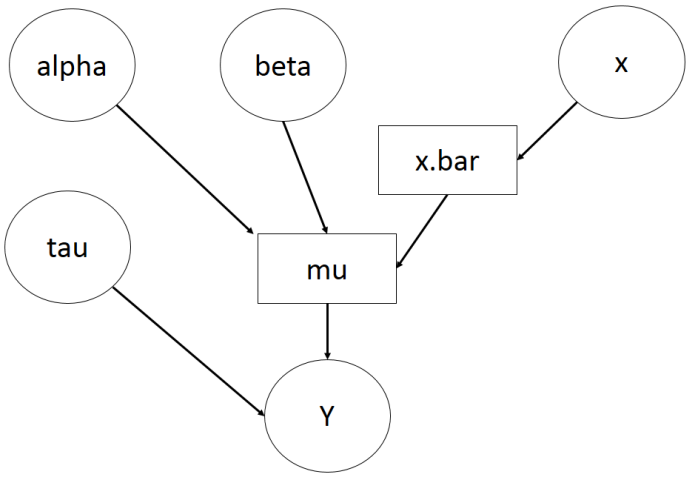

Figure 2: A probabilistic hierarchical model for the linear regression example of Section 3.

4 Markov Chains

A Markov chain is a model for a sequence of random variables, that is suitable to a class of stochastic processes that require to reduce the used amount of memory. Markov chains are often used to model sequences of observations in time, especially when the time variable is discrete.

Let's consider a sequence of random variables $\theta_1, \theta_2, \ldots, \theta_n$ that can take values from the same finite alphabet $O = \{o_1, o_2, \ldots, o_m\}$. This could be the

case, for example, of the Dow-Jones Industrial average. The trend for this quantity can be modelled supposing that, at several discrete time instants, a different random variable θ_t took a value among $O = \{Up, Down, Unchanged\}$.

Generally speaking, for a sequence of discrete random variables the joint probability is the following

$$P(\theta_1, \theta_2, \ldots, \theta_n) = P(\theta_1) \prod_{i=1}^{n} P(\theta_i | \theta_{i-1} \theta_{i-2} \ldots \theta_1)$$

which models the fact that a variable θ_k depends on all the preceding variables in the sequence. For example, in the case of a sequence of 3 variables

$$P(\theta_1, \theta_2, \theta_3) = P(\theta_1) P(\theta_2 | \theta_2) P(\theta_3 | \theta_2 \theta_1)$$

A sequence of random variables is said to form a *first–order* Markov chain if the probability of a variable in the sequence is conditioned only by the preceding variable. This means that

$$P(\theta_i | \theta_{i-1} \theta_{i-2} \ldots \theta_1) = P(\theta_i | \theta_{i-1})$$

and the formula for the joint probability becomes

$$P(\theta_1, \theta_2, \ldots, \theta_n) = P(\theta_1) \prod_{i=1}^{n} P(\theta_i | \theta_{i-1})$$

this equation is also known as the *Markov assumption*. Markov models are particularly indicated to model phenomena characterized by a sequence of observations in time, where each observation depends on the preceding one. Often, it is necessary to model relations with the m preceding variables instead of the first, which requires to use mth–order Markov chains, whose definition is easily deducible from the first–order one. In Gibbs sampling the first–order Markov assumption is sufficient. The probabilities $P(\theta_i | \theta_{i-1})$ are called *transition* probabilities. We can suppose that the variables θ_i are not distinct, thus the chain is made up of a sequence of variables taken from a finite set $\{s_1, s_2, \ldots, s_g\}$. The s_i variables are usually said the *states* of the Markov chain.

Thus, we can calculate the probability of state s_i to be the state at time t in the sequence. It is the overall probability to pass from a preceding state s_j to s_i. Note that s_j could be s_i itself, because the Markov chain could present consecutive repetitions of the same state in the sequence

$$P_t(s_i) = \sum_{j=1}^{g} P_{t-1}(s_j) P_t(s_i | s_j)$$

The above formula 'says' that the probability that s_i is at time t in the sequence is given by all the possible ways to reach s_i from any preceding s_j state. The formula also accounts for the probability that s_j was the previous state in the sequence. Note that the transition probability to pass from s_i to s_j can be different depending on the time instant. In order to understand the relationship between Gibbs sampling and Markov chains explained in the next Section, we need to introduce some definitions.

A Markov chain is said to be *homogeneous* or *stationary* if the transition probability does not depend on time. This means that $P_t(s_i|s_j)$ is simply $P(s_i|s_j)$ and depends only on the pair of states.

A Markov chain is said to have reached an *invariant* probability $\Gamma(s_i)$ over the states when this persists forever. In other words

$$\Gamma(s_i) = \sum_{j=1}^{g} \Gamma(s_j) P_t(s_i|s_j)$$

If the chain is also homogeneous then we can substitute P_t with P. A finite Markov chain always has at least one invariant distribution.

A Markov chain is *ergodic* if $P_t(s_i)$ converges to an invariant distribution for $t \to \infty$. Furthermore, this is required to happen regardless of the choice of initial probabilities $P_0(s_i)$. An ergodic Markov chain can have only one invariant distribution, called *equilibrium* distribution. In Gibbs sampling we search for ergodic Markov chains that converge to an invariant distribution, which is just the posterior probability distribution.

A Markov chain is *aperiodic* if the return to a state can occur at irregular times.

The chain is *irreducible* if it is possible to go from any state to any other state (not necessarily in one step).

A Markov chain is *recurrent* if for any given state i, if the chain starts at i, it will eventually return to i with probability 1.

The chain is *positive recurrent* if the expected return time to state i is finite, otherwise it is *null* recurrent.

The *ergodic theorem* states that if a Markov chain is aperiodic, irreducible and positive recurrent then it is ergodic [16]. This theorem is used to demonstrate that Gibbs sampling can be ergodic under some conditions. The extension of this explanation to the case of continuous variables is quite easy and the given definitions apply also to probability distributions. We leave other details about Markov chains to more specific documents [18, 25, 10].

5 Gibbs Sampling

In this Section we use the concepts defined so far in order to explain the Gibbs Sampling algorithm. The main aim of this method is to sample the posterior probability distribution

$$p(\bar{\theta}|\bar{y}) = \frac{p(\bar{y}|\bar{\theta})p(\bar{\theta})}{p(\bar{y})}$$

The idea is that, having samples from the posterior, we can use them for making predictions or for understanding which is the best choice for the $\bar{\theta}$ parameters given the real data. The main issue around posterior probability distribution is that it can be very difficult to draw samples from such density function, especially when it is not a standard statistical distribution. For standard distributions, in fact, there are several techniques to generate samples because the shape is well known [16, 14, 4].

Gibbs sampling uses Markov chains to draw samples from posterior densities [3, 22]. The aim is not to directly sample from the complete function, but from

the conditional distributions of the θ_i variables with respect to all the other variables (full conditionals): $p(\theta_i|\theta_1, \ldots, \theta_i - 1, \theta_i + 1, \ldots, \theta_n, \bar{y})$.

The justification for this will be more clear later. The procedure is iterative and it can be demonstrated that the higher is the number of iterations, the closer are the samples to the ones from the posterior density. This comes from the usage of Markov chains of samples, drawn from the full conditional probabilities. Usually, these chains are ergodically convergent to the posterior density values.

5.1 Gibbs sampling rationale

In order to understand how Gibbs sampling works, we first explain (i) how the samples from the full conditionals are linked to those from the posterior probability density, (ii) how to build the analytical form of a full conditional, (iii) how to build a Markov chain to sample the full conditionals.

The posterior probability density can be written in the following way, which comes from the Bayes' rule

$$p(\theta_1, \theta_2, \ldots, \theta_n|\bar{y}) = p(\theta_1|\theta_2, \ldots, \theta_n, \bar{y})p(\theta_2, \ldots, \theta_n|\bar{y})$$

the same rule is valid also for the other variables. This means that sampling each full conditional in turn, gives values that are proportional to the posterior distribution. Gibbs sampling uses this relation to suggest to sample iteratively each variable, leaving the other variables at their preceding state in time. When a full conditional is sampled in this way, a new value for the conditioned variable is picked and then immediately used to sample the other variables that have not been sampled yet. Each full conditional is usually easier to sample than the complete posterior density. Furthermore, it can hopefully have the form of a standard distribution, which is unlikely to happen for the complete posterior density.

Gibbs sampling is divided in two parts: the first aims at obtaining analytical forms for each full conditional. The second generates samples iteratively with a Markov approach, in which the samples tend towards the posterior density ones [12].

5.2 Obtaining analytical forms for full conditionals

The algorithm for obtaining the analytical forms for the full conditionals proceeds with the following steps

1. write the complete formula of the posterior probability;
2. pick one parameter θ_i;
3. from the formula of the posterior probability, drop all the factors that do not depend on θ_i. At this point, if we suppose that the other parameters are fixed, then we have obtained a formula for the full conditional of θ_i;
4. use automatic simplification procedures to figure out if the conditional probability can be reduced to a known distribution;
5. repeat from step 2 for all the parameters.

Remember that, in the case the relations among the variables had been expressed as a hierarchical model, the formula of the posterior probability density would be the one reported in Section 3.1. For example, in the case of 4 variables

$$p(\theta_1, \theta_2, \theta_3, \theta_4) = p(\theta_1)p(\theta_2)p(\theta_4|\theta_1, \theta_2)p(\theta_3|\theta_2, \theta_4)$$

where some of the terms can be interpreted as likelihoods (the ones in which real data are involved) and other ones as conditionals or priors.

For example, suppose that the analytical formula of the complete posterior probability was

$$p(\lambda_1, \lambda_2, \beta|\bar{y}, \bar{t}) = \prod_{i=1}^{2} \lambda_i^{(y_i-1)} e^{-(t_i+\beta)\lambda_i} \beta^9 e^{-20\beta}$$

where $\bar{y}$ and $\bar{t}$ are real observations and $\{\lambda_1, \lambda_2, \beta\}$ are random variables. It is evident that the formula cannot be reduced to a standard distribution. But, if we drop in turn the factors that depend either on the λ_i or on β we obtain

$$p(\lambda_1|\lambda_2, \beta, \bar{y}, \bar{t}) = \lambda_1^{(y_i-1)} e^{-(t_i+\beta)\lambda_1} = Gamma(y_i, t_i + \beta)$$

which holds also for λ_2, and

$$p(\beta|\lambda_1, \lambda_2, \bar{y}, \bar{t}) = \beta^9 e^{-20\beta} = Gamma(10, 20)$$

Thus, in this case the conditional distributions are much more easy to be sampled, because their distribution is well known. We will mention techniques to sample from the posterior distribution also in the cases in which this reduction is not possible.

5.3 Gibbs sampling algorithm

To illustrate the Gibbs sampling algorithm, let's address the case of 3 random variables $\{\theta_1, \theta_2, \theta_3\}$ and some real observations $\bar{y}$. The posterior probability distribution is $p(\theta_1, \theta_2, \theta_3|\bar{y})$ and we want to sample from it having the formulae for priors and likelihoods [26].

The steps of the Gibbs sampling algorithm (Gibbs sampler) are

1. pick a vector of starting values for the random variables from the prior distributions of the variables;

$$\theta^{(0)} = \{\theta_1^{(0)}, \theta_2^{(0)}, \theta_3^{(0)}\}$$

2. select θ_1 and draw a sample $\theta_1^{(1)}$ from the conditional probability of this variable, fixing the values of the other variables to $\theta_2^{(0)}$ and $\theta_3^{(0)}$. In other words, draw a sample from $p(\theta_1|\theta_2^{(0)}, \theta_3^{(0)}, \bar{y})$;

3. select θ_2 and draw a sample from $p(\theta_2|\theta_1^{(1)}, \theta_3^{(0)}, \bar{y})$, using the updated value of θ_1;

4. select θ_3 and draw a sample from $p(\theta_3|\theta_1^{(1)}, \theta_2^{(1)}, \bar{y})$, using both the previous updated values;

5. define the vector $\theta^{(1)} = \{\theta_1^{(1)}, \theta_2^{(1)}, \theta_3^{(1)}\}$

6. build a sequence of vectors $\theta^{(0)}, \theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(t)}$ by using the above sampling procedure, stopping at a certain $t^* = M$.

After a certain number of iterations, we will have M samples of the variables, where the last ones are the most reliable if the M value is 'big' enough. The above algorithm can be easily adapted to the case of more than 3 variables.

5.4 Gibbs sampling correctness

The Gibbs sampler produces a Markov chain, because at each step it considers each variable with respect to the others either at the previous time instant or at the current time instant. In other words, if the variables to sample are $\{\theta_1, \ldots, \theta_n\}$, then we can say that the Gibbs sampler builds the following transition probabilities distributions

$$p_t(\theta_i^{(t)} | \theta_1^{(t)}, \ldots, \theta_{i-1}^{(t)}, \theta_{i+1}^{(t-1)}, \ldots, \theta_n^{(t-1)}, \bar{y})$$

which are compliant with the definition of transition probability distribution of a Markov chain. The chain is also *invariant*, because for each variable the transition probability distribution fixes the other variables and forces the conditional distribution of $\theta_i^{(t)}$ to account for the likelihood to the data. In other words, the new sample is constrained to follow the conditional probability density (less than a normalization factor), because the full conditionals are factors of the expected posterior density. In fact the procedure starts from the 'desired' posterior density to obtain the full conditionals and this ensures that the samples will follow more and more the posterior, as far as new samples are iteratively produced. Furthermore, if the transition probabilities are all non-zero (i.e. the chain is irreducible), then the probability of remaining in the same state is non-zero. From this, It can be demonstrated that the hypotheses of the ergodic theorem are satisfied, thus in this case the chain is also ergodic and tends towards the posterior distribution for construction [16].

5.5 Producing optimal estimations from the samples

The expected value of a function of a random variable that can range between two numbers a and b is defined as

$$E[a(x)] = \int_a^b a(x)p(x)\mathrm{d}x$$

where $p(x)$ is the probability distribution of x. *Monte Carlo Integration* is a technique to approximate this integral by means of the samples of x [25, 15]. The technique estimates the expected value of a random variable with an average on the samples of x drawn from $p(x)$. Note that $p(x)$ could also indicate a posterior density. More formally

$$E[a(x)] \approx \frac{1}{n} \sum_1^n a(x_i)$$

where $\{x_1, \ldots, x_n\}$ are n samples of x drawn from $p(x)$.

In the same way, the expected value of x after the sampling can be approximated in this way

$$E[x] = \int_a^b xp(x)\mathrm{d}x \approx \frac{1}{n}\sum_1^n x_i = \tilde{\mu}$$

The variance of x is defined as $V[x] = E[(x - E[x])^2]$. Thus, it easy to demonstrate that we can approximate it as

$$V[x] \approx \frac{1}{n-1}\sum_1^n (x_i - \tilde{\mu})^2$$

The rationale under Monte Carlo Integration is that as the number of samples increases the approximation becomes more accurate. This statement is a direct consequence of the Strong Law of Large Numbers [13] if the samples are independent. Unfortunately, Gibbs sampling generates samples that are slightly dependent, but there are some good practices that allow to reduce this bias. As we have explained, the Markov chain produced by the Gibbs sampler possibly begins to converge after the generation of many samples. Thus, it is good practice to discard the first k produced values, where k depends on the speed of convergence of the chain. These are usually referred to as *burn–in* iterations. Furthermore, in order to break the dependency between the draws in the Markov chain, usually one draw every d is kept, where d is heuristically chosen. This practice is known as *thinning*. Finally, another good practice is to avoid sensitivity to the starting point of the chain. This is achieved by producing several Markov chains that start from different initial values and by sampling from all of them (*multiple chains*). Burn–in, thinning and multiple chains are standard initialization parameters for JAGS.

Monte Carlo integration can be applied to the Gibbs sampling output. In this case we take samples from the posterior probability distribution and the expected value for each variable is the average of the samples. This is a good practice if the sampling covers areas where most of the probability is concentrated. On the other side, when the distribution is strongly skewed or if it has a complex shape, it can be difficult to find such areas and the simple average on the samples cannot be appropriate. In such cases percentiles or mode are more appealing.

6 JAGS approach to Gibbs Sampling

JAGS applies Gibbs sampling in a transparent way to the user. The user is only asked to setup the probability density functions for priors, conditionals and likelihoods. As yet said, JAGS autonomously recognizes which are the likelihoods among the defined probability densities and the model is mapped onto a *hierarchical* model. After this phase, JAGS writes the formula for the posterior distribution and applies a simplification process to write the full conditionals for all the variables. Gibbs sampler is then applied, which sometimes combines other complex strategies to sample the full conditionals (e.g. Slice sampling or Adaptive Rejection sampling). Alternatively, JAGS could use the Metropolis–Hastings algorithm instead of the standard Gibbs sampling, in order to directly sample the posterior density.

Important input parameters to JAGS are (i) the number of Markov chains to produce (in order to possibly avoid non–ergodic behaviour), (ii) the burn–in iterations (to reduce the unreliability of the first samples), (iii) the thinning parameter (to enhance samples independency), (iv) indication on real observations (to enable likelihoods detection). JAGS produces the samples, then the user can apply Monte Carlo Integration to calculate the optimal values for the parameters, otherwise the user can calculate percentiles or other quantities. In the following, we propose a complete example of an R program, with comments in line, that uses JAGS with Monte Carlo Integration to get the optimal estimates for the parameters.

```
#True parameters
a = 10
b = 20
slope = b/a
#Build of the Theoretical Hockey-Stick function
numberOfRealdata = 100
#samples on the x axis
xsamples<-seq(length=numberOfRealdata, from=1, to=numberOfRealdata)
#samples of the theoretical H-S
theoretical_hockeyStick<-ifelse(xsamples < a, xsamples * slope, b)

#Let's add noise to the Theoretical Hockey-Stick function
noisy_hockeyStickSamples<-theoretical_hockeyStick+runif(numberOfRealdata,
    -1, 1);
#We now forget the theoretical Hockey-Stick and use the noisy samples to
    reconstruct it back

#We suppose that a, b and slope are indicative values for the best
    estimates
#We admit a high uncertainty around these indicative values
SD.a = 5
SD.b = 5
SD.slope = 10 #making the slope a random variable is not mandatory

#We initialize JAGS by stating which are the real data
jags.data <- list("N","a","b","slope","SD.a","SD.b","SD.slope",
              "xsamples","noisy_hockeyStickSamples")
#We state that we are interested into obtaining estimates for a,b and
    slope given the real data
jags.params <- c("random_a","random_b","random_slope")

#Let's build the BUGS model (the order of the following declarations is
    not important)
Model = "
model {
#static definitions
random_slopetau<-pow(SD.slope,-2)
random_a_tau <- pow(SD.a, -2)
random_b_tau <- pow(SD.b, -2)

#prior probabilities
random_slope ~ dnorm(slope, random_slopetau) #BUGS wants tau instead of
```

```
    sd directly
random_a ~ dnorm(a,random_a_tau)
random_b ~ dnorm(b,random_b_tau)

#likelihoods - JAGS will understand their nature from the fact that real
    data are available for them
#we state that each sample comes from a normal distribution around the
    random parameters values
for (j in 1:N){
   #definition
   y[j] <- ifelse(xsamples[j] < random_a, xsamples[j] * random_slope,
       random_b)
   #likelihood
   noisy_hockeyStickSamples[j] ~ dnorm (y[j], random_b_tau)
  }
}
"
# Write the BUGS model into a file
JAGSFILE="r2ssb.bug"
cat(Model, file=JAGSFILE)

#setup the Gibbs sampling
Nchains = 2 #number of Markov chains - to account for non ergodic
    convergence
Nburnin = 100#burn-in iterations - n. of iterations to discard
Niter = 1000#total n. of iterations
Nthin = 10#thinning - take every 10 samples to lower the dependency
    among the samples

#run the Gibbs sampling
jagsfit <- jags(data=jags.data, working.directory=NULL, inits=NULL,
    jags.params,
              model.file=JAGSFILE, n.chains=Nchains, n.thin=Nthin,
                  n.iter=Niter, n.burnin=Nburnin)

#recover the outputs
random_a_samples<-jagsfit$BUGSoutput$sims.list$random_a
random_b_samples<-jagsfit$BUGSoutput$sims.list$random_b
random_slope<-jagsfit$BUGSoutput$sims.list$random_slope

#Perform Monte Carlo Integration to recover the best estimates for a and
    b
a_best<- mean(random_a_samples)
SD.a_best<- apply(as.matrix(a_best),2,sd)
b_best<- mean(random_b_samples)
SD.b_best<- apply(as.matrix(random_b_samples),2,sd)

cat("Best estimate for a :",a_best, "\n")
cat("Best estimate for b :",b_best, "\n")

#plot the real values and the results
plot(xsamples,noisy_hockeyStickSamples,col="blue")
#plot the true value of a
abline(v=a, lty=3, col="red",lwd=1.5)
```

```
text(x=a+10, y=9,"True value of a")
#plot the re-estimated Hockey-Stick function
lines(x=c(0,a_best,numberOfRealdata), y=c(0,b_best,b_best),col="green")
text(x=a+40, y=17,"Re-estimated Hockey-Stick Function (line)")
```

The output of the above code is the following, along with the charts displayed in figure 3

```
>Best estimate for a : 9.74
>Best estimate for b : 19.9
```

By running the code several times it becomes evident that the re–estimated values are always very similar, even if the uncertainty on the prior expectations was high. This means that the produced Markov chain is ergodic.

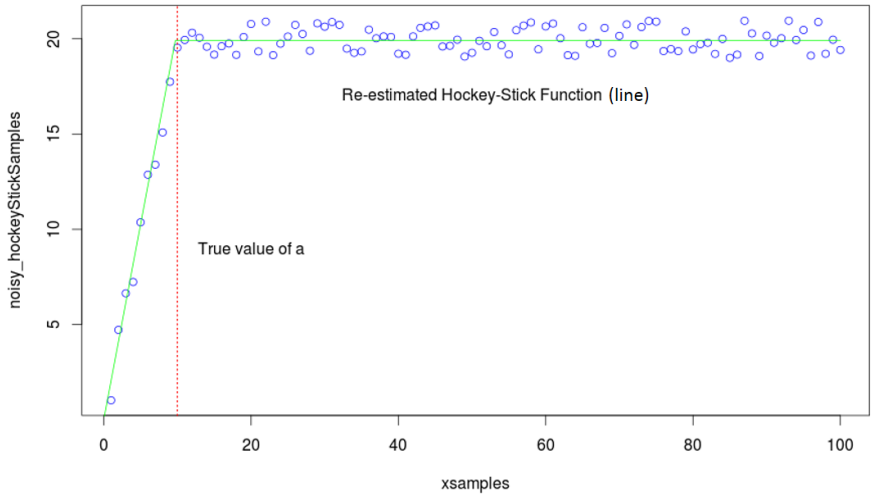

Figure 3: Output of the R example. The noisy observations, the true value of a and the re–estimated hockey-stick function are displayed.

7 Conclusions

In this document we have shown the basic principles of Gibbs sampling and its realization by means of the JAGS software. We have described the basic principles of Bayesian Inference and an approach that uses Markov chains and Monte Carlo Integration to estimate the expected values of the model parameters. The main aim has been to give some insight of these principles in order to make a JAGS user more aware of what happens behind the scenes. In the end of the document we have presented a complete R–code that uses JAGS to estimate a hockey–stick function. We suppose that the way the reader understands it is enriched with theoretical information about the underlying calculation mechanisms.

References

[1] James O Berger. *Statistical decision theory and Bayesian analysis*. Springer, 1985.

[2] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 1. Springer New York, 2006.

[3] George Casella and Edward I George. Explaining the gibbs sampler. *The American Statistician*, 46(3):167–174, 1992.

[4] Siddhartha Chib. Marginal likelihood from the gibbs output. *Journal of the American Statistical Association*, 90(432):1313–1321, 1995.

[5] Siddhartha Chib and Edward Greenberg. Understanding the metropolis-hastings algorithm. *The American Statistician*, 49(4):327–335, 1995.

[6] Aaron Clauset, Cristopher Moore, and Mark EJ Newman. Hierarchical structure and the prediction of missing links in networks. *Nature*, 453(7191):98–101, 2008.

[7] Gabriela Espino-Hernandez. Winbugs for beginners, 2010. http://www.stat.ubc.ca/lib/FCKuserfiles/WinBUGSforbeginners.pdf.

[8] Walter R Gilks, NG Best, and KKC Tan. Adaptive rejection metropolis sampling within gibbs sampling. *Applied Statistics*, pages 455–472, 1995.

[9] Soren Hojsgaard. A short introduction to using jags and r, 2013. http://people.math.aau.dk/~kkb/Undervisning/Bayes13/sorenh/docs/JAGS–intro–slides.pdf.

[10] Xuedong Huang, Alejandro Acero, Hsiao-Wuen Hon, et al. *Spoken language processing*, volume 15. Prentice Hall PTR New Jersey, 2001.

[11] PeterJ. Huber. Robust statistics. In Miodrag Lovric, editor, *International Encyclopedia of Statistical Science*, pages 1248–1251. Springer Berlin Heidelberg, 2011.

[12] Patrick Lam. Mcmc methods: Gibbs sampling and the metropolis-hastings algorithm, 2013. http://www.people.fas.harvard.edu/~plam/teaching/methods/mcmc/mcmc_print.pdf.

[13] Michel Loeve. Probability theory. *Graduate Texts in Mathematics*, 45:12, 1963.

[14] Soren Hojsgaard Claus Ekstrom Lyle Gurrin, Bendix Carstensen. Practical data analysis with jags using r, 2013. http://bendixcarstensen.com/Bayes/Cph-2012/pracs.pdf.

[15] David JC MacKay. Introduction to monte carlo methods. In *Learning in graphical models*, pages 175–204. Springer, 1998.

[16] Radford M Neal. Probabilistic inference using markov chain monte carlo methods, 1993.

[17] Radford M Neal. Slice sampling. *Annals of statistics*, pages 705–741, 2003.

[18] James R Norris. *Markov chains*. Number 2008. Cambridge university press, 1998.

[19] Ioannis Ntzoufras. *Bayesian modeling using WinBUGS*, volume 698. Wiley. com, 2011.

[20] Martyn Plummer. Jags: A program for analysis of bayesian graphical models using gibbs sampling. In *Proceedings of the 3rd International Workshop on Distributed Statistical Computing (DSC 2003). March*, pages 20–22, 2003.

[21] Martyn Plummer. rjags: Bayesian graphical models using mcmc. *R package version*, 2(0), 2011.

[22] Philip Resnik and Eric Hardisty. Gibbs sampling for the uninitiated. Technical report, DTIC Document, 2010.

[23] D Spiegelhalter, A Thomas, N Best, and D Lunn. Openbugs user manual, version 3.0.2. *MRC Biostatistics Unit, Cambridge*, 2007.

[24] Brenda Stefano. Using linear regression analysis and the gibbs sampler to estimate the probability of a part being within specification. *Quality and reliability engineering international*, 14(4):237–246, 1998.

[25] Brian Walsh. Markov chain monte carlo and gibbs sampling, 2004. http://web.mit.edu/ wingated/www/introductions/mcmc-gibbs-intro.pdf.

[26] Darren Wilkinson. Gibbs sampler in various languages, 2013. http://darrenjw.wordpress.com/2011/07/16/gibbs-sampler-in-various-languages-revisited/.