Gibbs Sampling

Prof. Dr. Narayan Prasad Adhikari
Central Department of Physics
Tribhuvan University Kirtipur, Kathmandu, Nepal

March 21, 2024

Assigned Problems

- Introduction
- Definition and Properties
- Implementation and optimization
- Forming the Sample
- Scanning strategies
- Using the sample
- Reparametrization
- Convergence diagnostics
- Applications

Introduction

- Gibbs sampling was originated in the context of image processing. In this context, the posterior of interest for sampling is a Gibbs distribution. Borrowing concepts from Mechanical Statistics, the density of the Gibbs distribution can be written as

$$f(x_1, x_2, ..., x_d) \propto \exp\left[\frac{-E(x_1, x_2, ..., x_d)}{kT}\right] \tag{1}$$

where k is a positive constant and T is absolute temperature. E is the energy of the system, a positive function, and x_i is the characteristic of interest for the ith component of the system, $i = 1, ..., d$. In Mechanical Statistics, x_i is the position or perhaps the velocity and position of the ith particle and in image processing it is (an indicator of) the colour of the ith pixel of an image.

■Introduction

- The energy function E is commonly given by a sum of potential functions V. These sums operate over collections of subgroups of components over which each potential function is evaluated.

- The subgroups generally obey some neighboring relationship in their definition. This leads to a probability specification based on local properties, useful for modelling spatial interaction between components. The main drawback is the difficulty in the determination o f the global properties, such as the normalizing constant.

- Gibbs sampling scheme could in fact be used for a host of other posterior distributions.

■In Physics

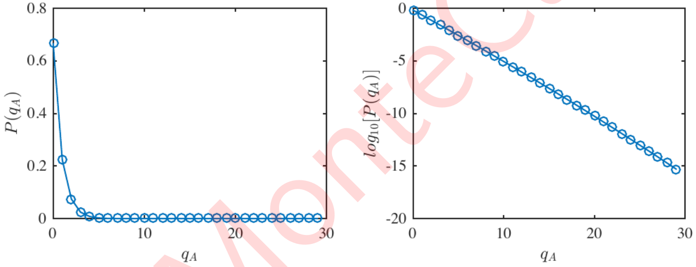


Figure: The Gibbs distribution gives how the energy is distributed in the system.

■Definition and Properties

- The Gibbs sampler is a very useful tool for simulations of Markov processes for which the transition matrix cannot be formulated explicitly
- Gibbs sampling is a Markov Chain Monte Carlo (MCMC) scheme where the transition kernel is formed by the full conditional distributions. Let us assume as before that the distribution of interest is $\pi(\theta)$ where $\theta = (\theta_1, ..., \theta_d)'$. Each one of the components θ_i can be a scalar, a vector or a matrix. However in our case we consider them as scalar.
- Consider also that the full conditional distributions $\pi_i(\theta_i) = \pi(\theta_i|\theta_{-i}),\ i = 1, 2, ..., d$ are available.
- This means that they are completely known and can be sampled from.

Definition and Properties

- The problem to be solved is to draw from π when direct generation schemes are costly, complicated or simply unavailable but when generations from the π_i, are possible.
- Gibbs sampling provides an alternative generation scheme based on successive generations from the full conditional distributions.
- **To carry on the Gibbas sampling we use following algorithm (See next page)**

■Definition and Properties -Algorithm

- Initialize the iteration counter of the chain $j = 1$ and set initial values

$$\theta^{(0)} = (\theta_1^{(0)}, \theta_2^{(0)}, ..., \theta_d^{(0)})'$$

- Obtain a new value $\theta^{(j)} = (\theta_1^{(j)}, \theta_2^{(j)}, ..., \theta_d^{(j)})'$ from $\theta^{(j-1)}$ through successive generations of values

$$\begin{aligned} \theta_1^{(j)} &= \pi(\theta_1 | \theta_2^{(j-1)}, \theta_3^{(j-1)}, ..., \theta_d^{(j-1)})' \\ \theta_2^{(j)} &= \pi(\theta_2 | \theta_1^{(j)}, \theta_3^{(j-1)}, ..., \theta_d^{(j-1)})' \\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ \theta_d^{(j)} &= \pi(\theta_d | \theta_1^{(j)}, \theta_2^{(j)}, ..., \theta_{d-1}^{(j)})' \end{aligned}$$

- Change counter j to $j + 1$ and return to above step until convergence is reached.

■Definition and Properties

- When convergence is reached, the resulting value $\theta(j)$ is a draw from π. As the number of iterations increases, the chain approaches its equilibrium condition. Convergence is then assumed to hold approximately.
- The obvious form to obtain a sample of size n from π is to replicate n chains until convergence.
- Alternatively, after convergence all draws from a chain come from the stationary distribution. Therefore n successive values from this chain after the burn-in period will also provide a sample from π .

■Algorithm - two variables case

- Data distribution:
- Assume that $y|\theta \sim N(\theta, \Sigma)$ is a bivariate normal distribution with unknown mean $\theta = (\theta_1, \theta_2)$ and known covariance matrix (in covariance matrix variances are given by diagonal elements whereas covariance is given by non-diagonal elements)

$$\Sigma = \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}$$

- Prior distribution: The prior for θ is an improper uniform over the real line i.e. $p(\theta_1, \theta_2) \propto 1$.
- Posterior distribution: Assuming we observe a single observation $y = (y_1, y_2)$,

$$\theta|y \sim N(y, \Sigma)$$

■Algorithm - two variables case

- Full conditional distribution:

$$\theta_1|\theta_2, y \sim N(y_1 + \rho(\theta_2 - y_2), 1 - \rho^2)$$
$$\theta_2|\theta_1, y \sim N(y_2 + \rho(\theta_1 - y_1), 1 - \rho^2)$$

  Sample from the posterior distribution using a Gibbs sample assuming $y = (0, 0)$ and $\rho = 0.8$

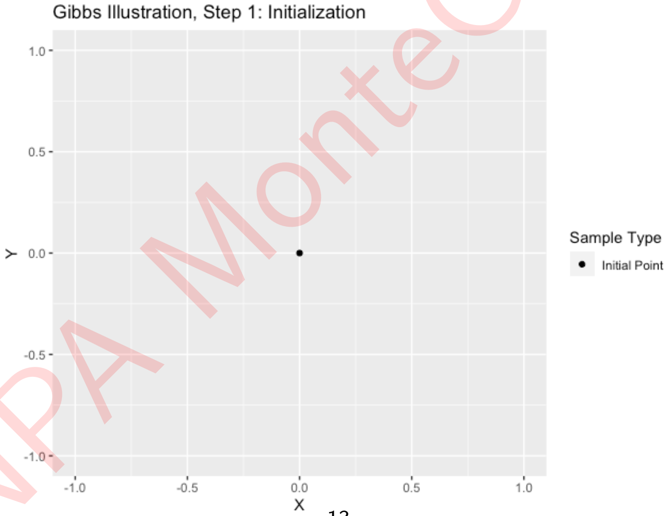
- **Algorithm and code**

■Algorithm - two variables case

- The gist of the Gibbs sampler is simple: sample from known conditional distributions, and use that resulting value to sample the next random variable from the following conditional probability distribution, ad infinitum.
- Algorithm:
- 1. Initialize (x_0, y_0) and set time n or t =0
- 2. Draw x_t from conditional distribution $X_t|(Y_{t-1} = y_{t-1} \sim N(\rho y_{t-1}, 1-\rho^2))$
- 3. Draw y_t from conditional distribution $y_t|(x_t = x_t \sim N(\rho x_t, 1-\rho^2))$
- 4. Increase t=t+1
- 5. Return to step 2

■Algorithm - two variables case

- Lets consider the case for $\rho = 0.9$ Step 1: Initialize $x_0 = 0.;\ y_0 = 0.$ also set the iteration counter n OR (t) to 0.



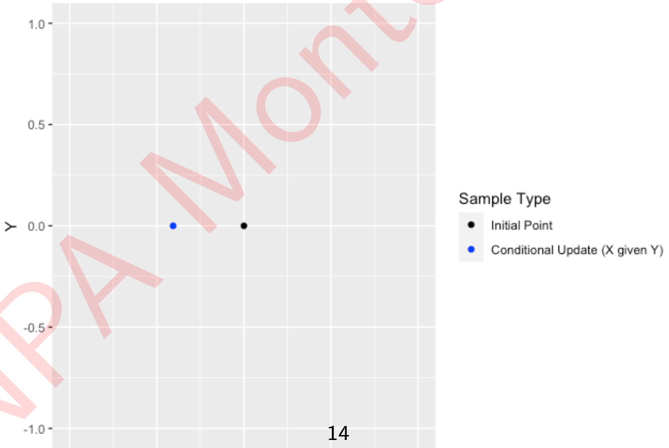
■Algorithm - two variables case

- Step 2: Conditional update of X given Y

$$X_1 |\ (Y_0 = 0) \sim N(0 \times \rho, 1 - \rho^2)$$

In one of the case it was -0.4 as shown below.

Gibbs Illustration, Step 2: Conditional Update for X given Y

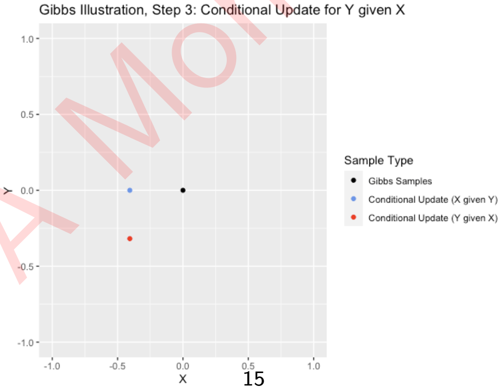


■Algorithm - two variables case

- Step 3: Conditional update of Y given X

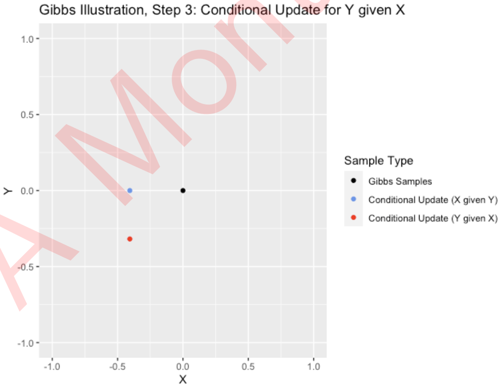
$$Y_1|\ (X_1 = -0.4) \sim N(-0.4 \times \rho, 1 - \rho^2)$$

In one of the case it was -0.32 as shown below. This time, the X coordinate of our new point is the same as the X coordinate of the point from step 2.



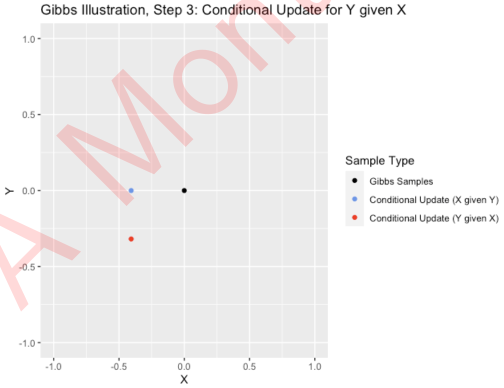
■Algorithm - two variables case

- Now we consider more steps repeating above processes. In this way one can sample the conditional probability of X given Y and vice versa. For multivariate systems its not easy to handle analytically. Even in this bivariate case for many steps its not possible to solve this problem analytically.

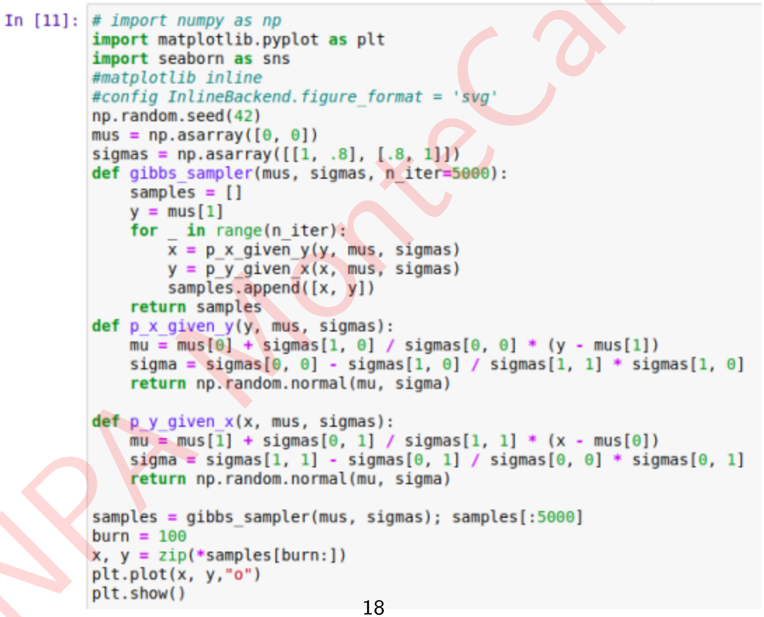


■Python code - two variables case

- Now we consider more steps repeating above processes. In this way one can sample the conditional probability of X given Y and vice versa. For multivariate systems its not easy to handle analytically. Even in this bivariate case for many steps its not possible to solve this problem analytically.



■Python code - two variables case

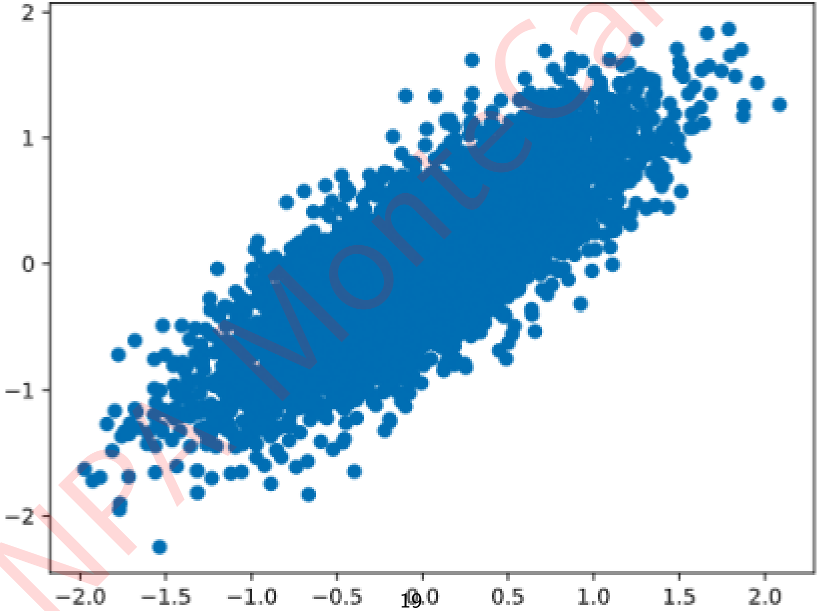


```
In [11]: # import numpy as np
         import matplotlib.pyplot as plt
         import seaborn as sns
         #matplotlib inline
         #config InlineBackend.figure_format = 'svg'
         np.random.seed(42)
         mus = np.asarray([0, 0])
         sigmas = np.asarray([[1, .8], [.8, 1]])
         def gibbs_sampler(mus, sigmas, n_iter=5000):
             samples = []
             y = mus[1]
             for _ in range(n_iter):
                 x = p_x_given_y(y, mus, sigmas)
                 y = p_y_given_x(x, mus, sigmas)
                 samples.append([x, y])
             return samples
         def p_x_given_y(y, mus, sigmas):
             mu = mus[0] + sigmas[1, 0] / sigmas[0, 0] * (y - mus[1])
             sigma = sigmas[0, 0] - sigmas[1, 0] / sigmas[1, 1] * sigmas[1, 0]
             return np.random.normal(mu, sigma)

         def p_y_given_x(x, mus, sigmas):
             mu = mus[1] + sigmas[0, 1] / sigmas[1, 1] * (x - mus[0])
             sigma = sigmas[1, 1] - sigmas[0, 1] / sigmas[0, 0] * sigmas[0, 1]
             return np.random.normal(mu, sigma)

         samples = gibbs_sampler(mus, sigmas); samples[:5000]
         burn = 100
         x, y = zip(*samples[burn:])
         plt.plot(x, y,"o")
         plt.show()
```

■Python code - two variables case



■Implementation and Optimization

Despite the theoretical results ensuring the convergence of the Gibbs sampler, its practical implementation may be complicated by the potential complexity of the models considered. Convergence of the sampler becomes difficult to characterize. Given that it is a numeric and iterative method, practical strategies to improve the efficiency of the method may have a considerable impact on its computational cost. Efficiency broadly consists of reducing the number of burn-in iterations and the amount of arithmetic operations required at each iteration. The techniques presented are related to the basic MCMC methods.

■Implementation and Optimization-Forming the sample

There are two forms to obtain a sample of size n from the posterior distribution π.

(i)The obvious one is to process$n$ chains in parallel until convergence, say after m iterations, and take as sample elements the mth chain value from each of the n chains. The generation procedure will then require mn generations from the chain. If chains are initialized independently, the sample consists of independent values from π. Independence is easier to establish if the initial values are all different and preferably with larger dispersion than in the posterior.

■Implementation and Optimization-Forming the sample

(ii) Another form is to consider a single chain and explore ergodic results. After convergence, all chain values have marginal distribution given by the equilibrium distribution π. So, a sample of size n may be formed by n successive values from this chain. This generation will require $m + n$ generations from the chain. This is substantially less than independent sampling. The difficulty here is that the sample elements are no longer independent due to chain dependence. Ergodic theorems ensure that inference based on this sample is still valid. From a practical point of view, there may be problems if the chain autocorrelation is too high and the sample is not large enough to acknowledge it. In these cases, chains may take too long to adequately cover the entire parameter space appropriately. As a result, some relevant regions may be underrepresented in the sample.

■Implementation and Optimization-Forming the sample

An alternative approach accom modating independence is to take for the sample chain values at every kth iteration after the burn-in period. Markovian processes only have first order dependence. As the lag between iterations increases, chain values become less and less correlated and are virtually independent for a large enough value of the lag k. A sample of size n with quasi-independent elements thus requires $m + kn$ generations from the chain. The value of k is typically smaller than m and again an improvement over independent sampling is obtained. There is no gain in efficiency. This procedure is advantageous if computer storage of values is limited.

■Implementation and Optimization-Forming the sample

There is no general agreement on the subject although it is generally agreed that running n parallel chains in practice is computationally inefficient and unnecessary. The main debate is whether a few parallel chains are needed. If the convergence properties of the chain are well understood then clearly a single chain suffices. As these characteristics are hard to obtain, prudence suggests that a few pilot parallel chains should be run. If they quickly settle around common values then a single chain can be safely used to extract a large sample for inference. Otherwise, there may be minor characteristics of the posterior distribution such as secondary modes far from the mode that require very large samples to be noticed.

■Implementation and Optimization-Scanning strategies

The Gibbs sampler described above involved a complete scan over the components. All iterations consisted of visits to update the components in the same deterministic order, typically $1 \to\ 2\ \to\ ...\ \to\ d$. There are many other possible scanning or updating strategies for visiting the components of θ. There are a few schemes for the purpose.

(i) Geman and Geman proved proved convergence to the joint distribution in a discrete setting for all visiting schemes that guarantee that all components are visited infinitely often when the chain is run indefinitely. The reversible Gibbs sampler where at each iteration each component is visited in a fixed order and then visited again in reversed order satisfies this property.

■Implementation and Optimization-Scanning strategies

(ii) Another scheme where an i.o. schedule is guaranteed draws a number i from {1, 2, ..., d} with fixed positive probabilities at each iteration and only updates the θ_i at that iteration. To make it more comparable with the deterministic scan, an iteration of these random scans can be defined by a collection of d such updates.

(iii) In another scheme we consider a random permutation scan where at each iteration a permutation of {1, 2, ..., d} is selected and components are visited in that order.

■Implementation and Optimization-Scanning strategies

Assume now that π is a multivariate normal distribution with precision matrix $\Phi = (\phi_{ij})$. For this setting, convergence for the deterministic scan is faster than for the random scan if Φ is tridiagonal ($\pi(\theta_i|\theta_{-i}) = \pi(\theta_i|\theta_{i-1}, \theta_{i+1})$, for all i) OR if Φ has nonnegative partial correlations ($\phi_{ij} \leq 0$). This result is particularly important because both dynamic amd hierarchical models lead to tridiagonal matrices if variances are known. Their results also indicate that more precise distri butions lead to faster convergence both for the deterministic and random scans.

■Implementation and Optimization- Using the sample

Whatever the scheme chosen for forming the sample, after it is used a sample of vectors $\theta_1, ..., \theta_n$ generated from the posterior distribution π is available. Assume also the more general case where these are successive values from a single Markov chain. A sample from the ith component of θ is given by $\theta_{1i}, ..., \theta_{ni}$. Marginal point or interval summaries of any real function $\psi = t(\theta)$ are estimated by their corresponding estimators based on the sample. This is always a consistent estimator by the ergodic theorem.

■Implementation and Optimization- Using the sample

The posterior mean of ψ is estimated by $\hat{E}(\psi) = \hat{\psi} = (1/n)\sum_{j=1}^{n}\psi_j$ where $\psi_j = t(\theta_j)$, $j = 1, ...n$. The posterior variance of ψ is similarly estimated by noting that

$$\sigma_\psi^2 = Var(\psi) = E(\psi^2) - [E(\psi)]^2.$$

The expectation value is obtained by $\bar{t} \to E_\pi[t(\theta)]$ as $n \to \infty$. Similarly the σ_ψ^2 is estimated by $\hat{\sigma}_\psi^2$ where

$$\hat{\sigma}_\psi^2 = \hat{E}(\psi^2) - [\hat{E}(\psi)]^2 = \frac{1}{n}\sum_{j=1}^{n}\left(\psi_j \hat{\psi}\right)^2 \qquad (2)$$

the sample variance.

■Implementation and Optimization- Using the sample

The marginal densities $\pi(\theta_i)$ can be estimated by (a smoothed version of) the histogram of sampled values of θ_i . Better estimators can be obtained by using conditional distributions. Recalling that

$$\pi(\theta_i) = \int \pi(\theta_i|\theta_{-i})\pi(\theta_{-i})d\theta_{-i}$$

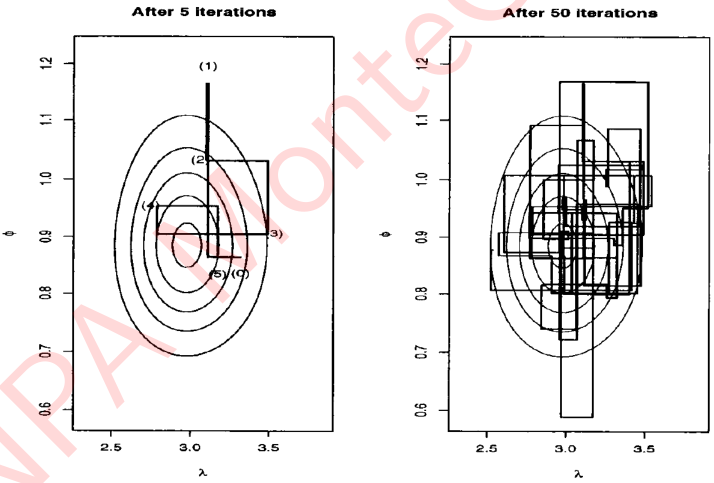
a Monte Carlo estimator is given by

$$\hat{\pi}(\theta_i) = \frac{1}{n}\sum_{j=1}^{n} \pi(\theta_i|\theta_{j,-i}) \tag{3}$$

where $\theta_{j,-i},\ j = 1, ..., n$ are samples from marginal $\pi(-\theta_i)$.

■Implementation and Optimization-Reparametrization

Consider the figure shown below.



■Implementation and Optimization-Reparametrization

- An iteration is formed by moves along the coordinate axes of the components of θ.

- If there is weak dependence between the components, the moves will be ample. Often, the posterior structure leads to high correlation between some of the components of θ
- Figure 2 illustrates this point for a bidimensional parameter. The contours o f the posterior show strong dependence between the components of θ and chain moves, governed by the conditional densities, will be small. The chain will take many iterations to adequately cover the parametric space and as a result convergence is slow. In this case, the Gibbs sampler will be inefficient. Examples can be constructed in larger dimension models where convergence can be slowed to any arbitrary amount of iterations

■Implementation and Optimization-Reparametrization

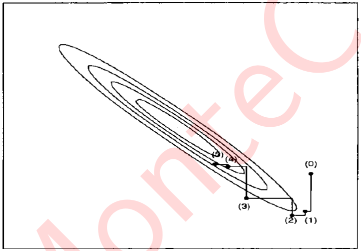


Figure: Contour lines of a bivariate posterior density with components highly correlated. A possible chain trajectory is also depicted to illustrate slow convergence, with iterations in parentheses. The contours are from a bivariate normal distribution with marginal distributions $\theta_1 \sim N(2,1)$ and $\theta_1 \sim N(3,1)$ and correlation -0.97. The trajectory is obtained by sampling from the full conditional distributions $\theta_2|\theta_1$ and $\theta_1|\theta_2$

Reparametrization- An Example

- We want to estimate the joint distribution of two variables, X and Y, where:
- X Uniform(-2, 2) (uniform distribution between -2 and 2)
- $Y = X^2$ (Y is simply the square of X)
- We face following Problem:
- While sampling X from a uniform distribution is straightforward, directly sampling Y from its conditional distribution (given X) is difficult. The typical Gibbs sampling approach would involve:

Reparametrization- An Example

- Sample X from its conditional distribution (which is easy in this case).
- Sample Y given the sampled X (which requires evaluating a complex function - squaring X).
- However, the issue here is that for most values of X, the corresponding Y value will be very close to 0 (since X^2 is small for small X). This creates a "bottleneck" effect around Y $=$ 0, making the chain mix slowly between high and low Y values.

Reparametrization- Algorithm of example

- One of the ways to solve above problem is Reparametrization
- To improve convergence, we can reparametrize Y and X both.
- Define the number of samples ($numsamples$).
- Initialize arrays
- Initialize arrays to store samples for both cases: with and without reparametrization
- $xsamplesnoreparam, ysamplesnoreparam$: Arrays to store samples without reparametrization.
- $xsamplesreparam$**,** $ysamplesreparam$**: Arrays to store samples with reparametrization.**

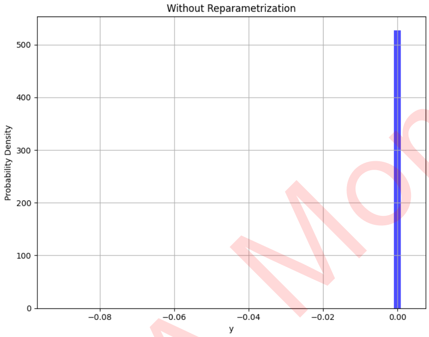
■ Reparametrization- Algorithm of example

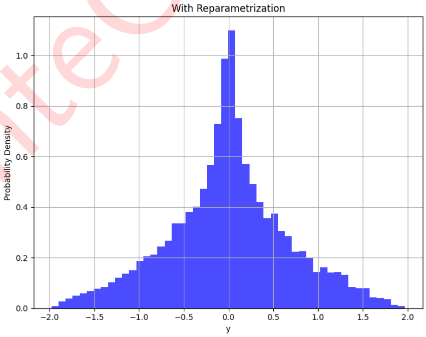
- Without Reparametrization:
- Initialize starting values for x and y using uniform distribution within the specified range.
- Implement Gibbs sampling iterations:
- i. Sample x from its conditional distribution given y.
- ii. Sample y from its conditional distribution given x.
- iii. Store the samples in the respective arrays.

Reparametrization- Algorithm of example

- With Reparametrization:
- Initialize starting values for u and v using uniform distribution within the range [0, 1].
- Reparametrize u and v to the range of x and y.
- Implement Gibbs sampling iterations:
- i. Sample u from its conditional distribution given v.
- ii. Sample v from its conditional distribution given u.
- iii. Reparametrize u and v to the range of x and y.
- iv. Store the samples in the respective arrays.
- plot all the data obtained.

■ Reparametrization- Algorithm of example





■Implementation and Optimization-Reparametrization

A simple and sometimes effective way to reduce convergence time is to use reparametrizations. Adequate transformations in the parameter space may produce situations of near independence that are ideal for fast convergence of the chain. Unfortunately, there are no rules to determine suitable transformations but frequently linear transformations that produce a diagonal variance matrix provide good results.

■ Reparametrization - An example

Random Effects (Hierarchial) model

consider the simple random effects model:

$$Y_{ij} = \mu + \alpha_i + \epsilon_{ij}$$

with i=1,..,I; and j=1,...,J where $\alpha_i \sim N(0, \sigma_\alpha^2)$ and $\epsilon_{ij} \sim N(0, \sigma_y^2)$.
For a flat prior on μ, the Gibbs sampler implemented for the $(\mu, \alpha_1, \ \dots \ \alpha_I)$ parametrization exhibits high correlations and consequent slow convergence if $\sigma_y^2/(IJ\sigma_\alpha^2)$ is large.
On the other hand, if the model is rewritten as the hierarchy

$$Y_{ij} \sim N(\eta_i, \sigma_y^2), \eta_i \sim N(\mu, \sigma_\alpha^2)$$

, the correlations between the η_i's and between μ and the η_i's are lower so converges faster than before.

■ Sampling from the full conditional distributions

In some cases, the form of the full conditional distribution is not recognizable which prevents sampling via the conventional algorithms.

Ritter and Tanner developed yet another sampling scheme from difficult full conditionals. Their approach is similar to adaptive rejection by being based on the evaluation of the full conditional at a few selected points. For that reason, they called it the griddy Gibbs sampler. Let $\pi_i(\theta_{ij})$, be a difficult full conditional distribution. Then, sampling from π_i can be approximately performed as follows:

■ Sampling from the full conditional distributions

1. Take a grid of points $\theta_{i1}, ..., \theta_{im}$, evaluate $\pi_i(\theta_{ij})$, j=1,...m and normalize them to obtain weights $w_1, ..., w_m$.
2. Use the weights $w_1, ..., w_m$ to construct a simple approximation to the distribution function of π_i.
3. Draw a value from π_i by the probability integral transform method.

■ Convergence Diagnostics

A value from the distribution of interest π is only obtained when the number of iterations of the chain approaches infinity. In practice this is not attainable and a value obtained at a sufficiently large iteration is taken instead of being drawn from π . The difficulty is the determination of how large this iteration should be. There is no simple answer to this question and most efforts have been directed at studying as close as possible the convergence characteristics of the chain.

Convergence Diagnostics

- There are two main ways to approach the study of convergence.

- The first one is more theoretical and tries to measure distances and establish bounds on distribution functions generated from a chain. In particular, one can study the total variation distance between the distribution of the chain at iteration j and the limiting distribution π. Special aspects derived from the probabilistic structure of the chain can also be studied.

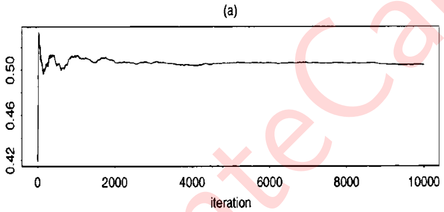
Convergence Diagnostics

- Another approach is statistical perspective i.e., by analyzing the properties o f the observed output from the chain. This is an empirical as opposed to a theoretical treatment of the problem and is obviously more practical. The difficulty with this approach is that it can never guarantee convergence because it is only based on observations from the chain
- The first one is more difficult than the second one.

Convergence Diagnostics

- One can observe the trajectory of a chain exhibiting the same qualitative behavior through iterations after a transient initial period is an indication of convergence. Similarly, the trajectory of the ergodic averages can be evaluated and plotted. An asymptotic behavior over many successive iterations indicates convergence.
- Figure next page shows the erdogic averages of variance components in a model data.

Convergence Diagnostics



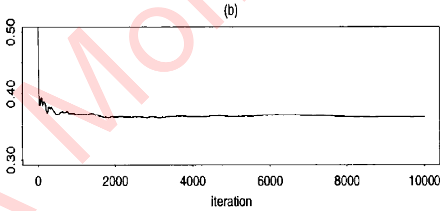


Figure: Ergodic averages of two parameters with number of iterations of the chain. The parameters are standard deviations of random effects at: (a) individual; (b) unit level in a longitudinal study of epilepsy treatment

■ Convergence Diagnostics

These, graphical, techniques must be used with caution and should always be accompanied by some theoretical reasoning. Graphical techniques may be deceptive indicating constancy that may not be so evident under a different scale. More importantly, there are many chains that exhibit every indication of convergence without actually achieving it. They are called metastable chains and are the subject of much research in probability theory.

Convergence Diagnostics- prescription

- Raftery and Lewis proposed a method to establish the length of a chain required for a M CM C run. More specifically, the methodology suggests values of m, the number of burn-in iterations, k, the number of iterations to be skipped between stored chain values and n, the size of the sample values that must be stored to achieve a given Monte Carlo precision of estimates.

Convergence Diagnostics- prescription

- The setting for these choices is the estimation of u, the q quantile (percentile) of a given function $\psi = t(\theta)$, i.e. $q = Pr_{\pi}(\psi \leq u)$. The method requires that the Monte Carlo estimate $\hat{q}$ satisfies $Pr(|\hat{q} - q| \leq r) = s$. A common choice is the tail probability with $q = 0.025$ in which case u is the lower limit of the equal tail 95% posterior credibility interval for ψ. One may require that the value o f this probability be estimated in a MCMC run with error smaller than r = 0.01 with confidence s = 0.99. So, 95% posterior intervals would be given by intervals with posterior probabilities between 93% and 97% with 99% confidence. This confidence level is due to the estimation of q by MCMC and should not be confused with posterior uncertainty about ψ, governed by π.

■ Convergence Diagnostics- Formal methods

- The methods presented here diagnose convergence based on exploration of the statistical properties of the observed chain. The methods here attempt to decide whether convergence can be safely assumed to hold rather that prescribing the run length to achieve convergence. There have been many methods presented in the literature. We consider only *Time series analysis.*

Convergence Diagnostics- Formal methods

- Consider a real function $\psi = t(\theta)$ and its trajectory $\psi^{(1)},\ \psi^{(2)},$ ... obtained from $\psi^{(j)} = t(\theta^{(j)}),\ j = 1, 2, \ldots$ This trajectory defines a time series and ergodic averages of this series can be evaluated.
- Geweke suggested the use of tests on ergodic averages to verify convergence of the chain based on the series $\psi^{(j)}$.

■ Convergence Diagnostics- Formal methods

- Assume observation of the chain for to $m + n$ iterations and form averages

$$\bar{\psi}_b = \frac{1}{n_b} \sum_{j=m+1}^{m+n_b} \psi^{(j)} \tag{4}$$

and

$$\bar{\psi}_a = \frac{1}{n_a} \sum_{j=m+n-n_a+1}^{m+n} \psi^{(j)} \tag{5}$$

where $n_b + n_a < n$.

- If m is the length of the burn-in period, then ψ_a and ψ_b are the ergodic averages at the end and beginning of the convergence period and should behave similarly. As n gets large and the ratios n_a/n and n_b/n remain fixed then

■ Convergence Diagnostics- Formal methods

$$z_G = \frac{\psi_a - \psi_b}{\sqrt{\hat{V}ar(\psi_a) + \hat{V}ar(\psi_b)}} \rightarrow N(0,1) \tag{6}$$

So, the standardized difference z_G between the ergodic averages at the beginning and at the end of the convergence period should not be large if convergence has been achieved. Large differences indicate lack of convergence but small differences do not imply convergence. Geweke suggested the use of values $n_b = 0.1n$ and $n_a = 0.5n$ and used spectral density estimators for the variances. This is a univariate technique.

■ Applications of Gibbs Sampling: Hierarchical Model

- Hierarchical models have the following structure - first we specify that the data come from a distribution with parameters θ

  $$X \sim f(X|\theta)$$

  and that the parameters themselves come from another distribution with hyperparameters λ

  $$\theta \sim g(\theta|\lambda)$$

  and finally that λ comes from a prior distribution

  $$\lambda \sim h(\lambda)$$

- More levels of hierarchy are possible - i.e you can specify hyper-hyperparameters for the distribution of λ and so on.

■ Applications of Gibbs Sampling: Hierarchical Model

- The essential idea of the hierarchical model is because the θs are not independent but rather are drawn from a common distribution with parameter λ, we can share information across the θs by also estimating λ at the same time.

- As an example, suppose we have data about the proportion of heads after some number of tosses from several coins, and we want to estimate the bias of each coin. We also know that the coins come from the same mint and so might share some common manufacturing defect.

■ Applications of Gibbs Sampling: Hierarchical Model

- There are two extreme approaches - we could estimate the bias of each coin from its coin toss data independently of all the others, or we could pool the results together and estimate the same bias for all coins. Hierarchical models provide a compromise where we shrink individual estimates towards a common estimate.
- Note that because of the conditionally independent structure of hierarchical models, Gibbs sampling is often a natural choice for the MCMC sampling strategy.

■ Applications of Gibbs Sampling: Hierarchical Model

- Suppose we have data of the number of failures (y_i) for each of 10 pumps in a nuclear plant. We also have the times (i) at which each pump was observed. We want to model the number of failures with a Poisson likelihood, where the expected number of failure λ_i differs for each pump. Since the time which we observed each pump is different, we need to scale each λ_i by its observed time t_i

- We now specify the hierarchical model - note change of notation from the overview above - that θ is λ (parameter) and λ is β (hyperparameter) simply because λ is traditional for the Poisson distribution parameter.

■ Applications of Gibbs Sampling: Hierarchical Model

The likelihood f is

$$\prod_{i=1}^{10} Poisson(\lambda_i \, t_i) \tag{7}$$

we let the prior g for λ be

$$\lambda \sim Gamma(\alpha, \beta) \tag{8}$$

with $\alpha = 1$ and let β to be a random variable to be estimated from the data

$$\beta \sim Gamma(\gamma, \delta) \tag{9}$$

with $\gamma = 0.01$ and $\delta = 1$.

There are 11 unknown parameters (10 λs and β) in this hierarchical model.

■ Applications of Gibbs Sampling: Hierarchical Model

The posterior is

$$p(\lambda, \beta | y, t) = \prod_{i=1}^{10} Poisson(\lambda_i \ t_i) \times Gamma(\alpha, \beta) \times Gamma(\gamma, \delta) \tag{10}$$

with the conditional distributions needed for Gibbs sampling given by

$$p(\lambda_i | \lambda_{-i}, \beta, y, t) = Gamma(y_i + \alpha, t_i + \beta) \tag{11}$$

and

$$p(\beta | \lambda, y, t) = Gamma(10\alpha + \gamma, \delta + \sum_{i=1}^{10} \lambda_i) \tag{12}$$

HW: Write Algorithm and hence code to solve above problem. You can take data as follows: