Markov Chain Monte Carlo Sampling

Bachelors Thesis Project

Bachelors of Technology
in
Computer Science and Engineering

by

Saurabh Garg
(140070003)

under the guidance of

Prof. Suyash Awate

Department of Computer Science and Engineering
Indian Institute of Technology Bombay

November, 2017

Abstract

Markov chain Monte Carlo (MCMC) methods are a class of algorithms for sampling from a probability distribution based on constructing a Markov chain that has the desired distribution as its equilibrium distribution. The state of the chain after a number of steps is then used as a sample of the desired distribution. The quality of the sample improves as a function of the number of steps.

Though a Markov chain gives us a neat solution, the problem with using a Markov chain is that it converges to its stationary distribution on when its simulated for a long time. To tackle this problem, the concept of perfect sampling was introduced which can tell if the the chain we are simulating has converged or not.

In this report, we would be getting an idea about Markov chains first and then go through various MCMC sampling methods in existence. We then see how perfect sampling fits into some of these methods and extend it over posteriors. We extend perfect sampling algorithms to sample from general posteriors using bounding chains. We also see, how we can use the posterior sampling methods for MRI Image Segmentation and uncertainty estimation.

Declaration

I declare that this written submission represents my ideas in my own words and where other's ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Date: 27th November, 2017 Saurabh Garg

Place: IIT Bombay, Mumbai Roll No: 140070003

Acknowledgements

This report is an outcome of my work under the guidance of Prof. Suyash Awate on Markov Chain Monte Carlo Sampling. I am thankful to the people who have been instrumental in helping me out throughout this Research project. First and foremost, I would like to express my sincere gratitude towards my supervisor Prof. Suyash Awate for his guidance. My research in Markov Chain Monte Carlo methods for sampling in higher dimensional space is being driven by his vision and support. It was immensely helpful in gaining the understanding required for writing this report.

I would also like to thank Krishna Harsha for brilliant ideas, for healthy and useful discussions on the topic.

Contents

Chapter 1

Introduction

Markov process, is a stochastic process that satisfies the Markov property (or memory-less property) i.e. if the conditional probability distribution of future states of the process (conditional on both past and present states) depends only upon the present state, not on the sequence of events that preceded it. Monte Carlo methods are a class of computational algorithms that rely on repeated random sampling to obtain numerical results. The aims of Monte Carlo methods are to solve these problems:
Problem 1: to generate samples $\{x^{(r)}\}_{r=1}^{R}$ from a probability distribution $\phi(x)$.
Problem 2: to estimate expectations of functions under this distribution.
Sometimes, the distributions are such that it is not easy to generate samples which are distributed exactly in the same way as the distribution we want to sample from. These samples points some times are not the exact representation of the underlying distribution. Lets say that we want to draw samples from the distribution $P(x)$ which we know with in a multiplicative constant such that

$$P(x) = P^*(x)/\mathcal{Z}$$

where $P(x)$ is the actual probability of occurring of x and (Z) is the normalization constant i.e. $(Z) = \int_X P^*(x)x$ where X is the entire space. Many methods require the value of $\mathcal{K}$ for correct sampling. In case of Gaussian, finding this $\mathcal{K}$ is very easy, but in general this can be a very tedious task. Now, consider a binary image of size 256×256, the number of operations required to find the value of $\mathcal{K}$ requires number of operations proportional to the number of total states possible i.e $2^{256\times256} \approx 10^{19730}$. Given the current computation power available, it is almost next to impossible to iterate over the state space and find the exact value of $\mathcal{K}$.

Markov chain Monte Carlo (MCMC) methods are a class of algorithms for sampling from a probability distribution based on constructing a Markov chain that has the desired distribution as its equilibrium distribution. The idea behind these methods is, if Markov chains run for long enough, their state will be distributed as their stationary distribution. Markov chains gives a

method to sample from certain distributions which otherwise would have been very difficult to sample from because their partition function can not be computed. Though Markov chains gives us a nice way, to sample from the distribution, the time of convergence is unknown i.e. the time it will take to converge to a stationary distribution. To overcome this problem, the idea of perfect sampling was introduced which can tell whether the chain which is being simulated has converged or not.

1.1 Organization of the report

In the next chapter, we will discuss some preliminary knowledge that is required to under stand the Markov chains and perfect sampling. In third chapter, we will introduce Markov chain Monte Carlo methods for sampling. In chapter four, we will discuss the idea of perfect sampling and few perfect sampling algorithms. We will also discuss, bounding chains algorithm for anti ferromagnetic system (repelling spin system) and sampling from posteriors. We will show, some results on posterior sampling and how it can be used to get exact MRI Brain Image segmentation.

Chapter 2

Prerequisites

In this chapter, we will discuss some sequence of sophisticated Monte Carlo Methods [8]. We will also discuss some basic properties of Markov chains.

2.1 Monte Carlo Methods

2.1.1 Importance Sampling

Importance sampling is a method to approximate the expectation of the function $\phi(x)$. Lets say, that the target distribution is $P(x)$ and we are able to evaluate it within a multiplicative constant $\mathcal{K}$, i.e. we have $P^*(x)$ such that $P(x) = P^*(x)/\mathcal{K}$. But $P(x)$ is very complicated to directly sample from, and we assume we know a simple density $Q(x)$ from which we can generate samples and which we can evaluate within in a multiplicative constant. In importance sampling, we generate N samples $\{x^{(n)}\}_{n=1}^{N}$ from $Q(x)$. To take in to account the fact that samples are from $Q(x)$ not from $P(x)$, we introduce weights

$$w_n = \frac{P^*(x^{(n)})}{Q^*(x^{(n)})}$$

which is used as the importance of each sample.

2.1.2 Rejection Sampling

Same as above we assume, we have $P(x)$ which is very complicated to directly sample from, and we assume we know a simple density $Q(x)$ from which we can generate samples and they can be

evaluated within in a multiplicative constant $\mathcal{K}$. We further assume that we know the value of constant c such that

$$cQ^*(x) \geq P^*(x) \quad \forall x$$

We generate two random numbers. The first, x, is generated from the proposal density $Q(x)$. We then evaluate $cQ(x)$ and generate a uniform random number u from the interval $[0, cQ(x)]$. Evaluate $P^*(x)$ and accept or reject the sample x by comparing the value of u with the value of $P(x)$. If $u \geq P(x)$ then x is rejected, otherwise it is accepted.

2.2 Markov Chains

Definition. *A discrete time stochastic process $X_n \mid n \in I$ with a countable state space S defined on a probability space (σ, F, P) is said to be a Markov chain if it satisfies the Markov property which is given as below*

$$P\{X_n = j | \{X_i | i \leq n, i \in I\}\} = P\{X_n = j | X_{n-1}\}, \quad \forall j \in S, n \in I$$

A Markov chain $\{X_n | n \in I\}$ is said to be time homogeneous if,

$$P\{X_n = j | X_{n-1} = i\} = P\{X_1 = j | X_0 = i\}, \quad \forall i, j \in S, n \in I$$

For a time homogeneous Markov chain $\{X_n | n \in I\}$, the transition matrix of the chain is defined as,

$$T = (t_{ij}), t_{ij} := P\{X_1 = j | X_0 = i\}, \quad \forall i, j \in S$$

2.3 Properties

2.3.1 Reducibility

Definition. *A Markov chain is said to be irreducible if it is possible to reach every state from every state.* Formally, this can be written as,

$$i \to j \qquad \forall i, j \in S$$

A state j is said to be accessible from a state i (written as i $\to$ j), if a system started in state i has a non-zero probability of transitioning into state j at some point. Formally, Formally, state j is accessible from state i if there exists an integer $n_{ij} \geq 0$ such that,

$$P(X_{n_{ij}} = j | X_0 = i) = p_{ij}^{(n_{ij})} \geq 0$$

where $p_{ij}^{(n_{ij})}$ is the value of the i,j^{th} element in the matrix $T^{n_{ij}}$. This gives the probability of going from state i to state j in n_{ij} steps.

2.3.2 Periodicity

Definition. *A Markov chain is aperiodic if every state is aperiodic.*
A state i has period k if any return to state i must occur in multiples of k time steps. Formally, the period of a state is defined as,

$$k = gcd\{n \geq 0 | P(X_n = i | X_0 = i) \geq 0\}$$

provided that this set is not empty. Oth- erwise the period is not defined. If k = 1, then the state is said to be aperiodic: returns to state i can occur at irregular times. It can be demonstrated that a state i is aperiodic if and only if there exists n such that for all $n' \geq n$,

$$P(X_{n'} = i | X_0 = i) > 0$$

2.3.3 Transience and Recurrence

Definition. *A state i is said to be transient if, given that we start in state i, there is a non-zero probability that we will never return to i.* Formally, let the T_i be the first return time to state i,

$$T_i = inf\{n \geq 1 | X_n = i | X_0 = i\}$$

The number

$$f_{ii}^{(n)} = P(T_i = n)$$

is the probability that we return to the state i for the first time after n steps. Therefore, i is transient if,

$$P(T_i < \infty) = \sum_{n=1}^{\infty} f_{ii}^{(n)} < 1$$

State i is recurrent if it is not transient.

2.3.4 Ergodicity

Definition *A state i is said to be ergodic if it is aperiodic and recurrent.* i.e. a state is ergodic, if it is recurrent, has a period of 1 and has a finite mean recurrence time. If all state are in an irreducible Markov chain is ergodic, then the chain is said to be ergodic.

2.3.5 Stationary Distribution

Definition π *is called the stationary distribution and the Markov chain with transition matrix T.*
Let $\{X_n | n \geq 0\}$ be a Markov chain with the state space S and transition matrix $T = (t_{ij})$. Let μ_0

denote the initial distribution. Let μ_n denote the distribution of X_n i.e. μ_n is a probability measure given by,

$$\mu_n(i) = P(X_n = i), \quad \forall i \in S$$

It can be seen that, $\mu_{n+1} = \mu_n P$. Suppose we know that $\mu_n \to \pi$, i.e., //

$$\forall \epsilon > 0, \quad \exists n_0 \quad s.t. \quad |\mu_n(i) - \pi(i)| < \epsilon, \quad \forall n \geq n_0, \quad i \in S$$

When $n \to \infty$ we get,

$$\pi = \pi T$$

and if $\mu_0 = \pi$, then,

$$\mu_1 = \mu_0 T = \pi T = \pi$$

$$\mu_n = \pi, \quad \forall n \geq 0$$

i.e. if the initial distribution of the Markov chain is given by π which satisfies $\pi = \pi T$, then the distribution of X_n remains as π i.e. it always remains stationary.

Theorem. *Let $X_n | n \geq 0$ be an irreducible, aperiodic Markov chain whose state space S is finite. Then,*

$$\lim_{n \to \infty} \sup |P\{X_n = j | X_0 = i\} - \pi(j)| = 0, \quad \forall i, j \in S$$

where π is the unique stationary distribution.

This is the basic fact that is exploited when constructing chains during the Monte Carlo Markov Chain procedure. This result tells us that as we tend to infinity, the distribution of our chain reaches that of the stationary distribution. It means that, as we run our chain from some initial distribution, if we run the chain for long enough, we will reach the stationary distribution.

2.3.6 Reversibility

Definition. *A Markov chain $\{X_n | n \geq 0\}$ is said to be reversible if there is a stationary distribution ÏĂ over its states such that:*

$$\pi_i t_{ij} = \pi_j t_{ji} \quad \forall i, j \in S$$

This is known as the detailed balance condition.

Reversible Markov chains are common in Markov chain Monte Carlo approaches because the detailed balance equation for a desired distribution π necessarily implies that the Markov chain has been constructed so that π is a steady-state distribution.

2.3.7 Coupling

Definition. *Coupling of Markov chains is a process $\{(X_n, Y_n) | n \geq 0\}$ such that $\{X_n | n \geq 0\}$ and $\{Y_n | n \geq 0\}$ are Markov chains with transition matrix P which may have different initial distributions such that*

$$X_{n+k} = Y_{n+k}, \quad k \geq 1 \quad \text{if } X_n = Y_n$$

Chapter 3

Markov Chain Monte Carlo Sampling Methods

One of the drawbacks of Importance sampling and rejection sampling is the proposed density $Q^*(x)$ is similar to $P^*(x)$. In many problems it is very difficult to create a density $Q(x)$ which is similar to $P^*(x)$ and yet simple.
Let π be the stationary distribution. Consider the problem of sampling from π and as we know from the properties of Markov chains mentioned above, that we sample from π, if we have a transition matrix T such that the reversibility condition is satisfied on T and π.

3.1 Metropolish Hastings Method

This algorithm [4], instead make use of a proposal density Q which depends on the current state $x^{(t)}$. The proposal density is $Q(x'; x^{(t)})$ is fixed density from which we can draw samples. We see $Q(x', x^{(t)})$ as the probability of transitioning from $x^{(t)}$ to a new state x', thus if it satisfies the reversibility condition then we are done as we got our transition matrix T. But, in general, this will not be the case. With out loss of generality assume,

$$\pi_x Q(x; y) > \pi_y Q(y; x)$$

To overcome this problem, we define a function $\alpha(.,.)$ such that the transition matrix is give by,

$$\alpha(x, y) = \min\{1, \frac{\pi_y Q(y; x)}{\pi_x Q(x; y)}\}$$

This gives a transition matrix which can be used to simulate a Markov chain whose stationary distribution is given by π. $\alpha(x; y)$ is interpreted as the probability of acceptance of new state.

Algorithm 1 Metropolish Hastings Algorithm

1: Initialise $x^{(0)}$ to some state $\in$ S
2: **for** i=1:N **do**
3: Generate y from $Q(.; x^{(i)})$ and u from $U(0,1)$
4: **if** $u < \alpha(x^{(i)}, y)$ **then**
5: $x^{(i+1)} \leftarrow y$
6: **else**
7: $x^{(i+1)} \leftarrow x^{(i)}$

3.2 Gibbs Sampling

Gibbs sampling [2] is generally used when we have more than one dimensions to sample from. Let, the state space is d-dimensional $x_1, x_2, x_3 ....., x_d$ and we wish to sample from distribution π which defined over this state space. Gibbs sampling is a special case of the Metropolish Hastings method. It is used when we can easily get the conditional distribution of one parameter conditioned on the other parameters. Thus, this the underlying assumption in the Gibbs Sampling algorithm and one iteration is given as,

$$x_1^{t+1} \sim P(x_1|x_2^t, x_3^t, ...x_d^t)$$
$$x_2^{t+1} \sim P(x_2|x_1^{t+1}, x_3^t, ..., x_d^t)$$
$$.$$
$$.$$
$$.$$
$$x_d^{t+1} \sim P(x_d|x_1^{t+1}, x_2^{t+1}, ..., x_{d-1}^{t+1})$$

This can be considered as a special case of Metropolish Hastings algorithm where the

$$\begin{aligned} Q(x,y) &= P(y_k|x_1, x_2..., x_{k-1}, x_{k+1}, ...x_d) \\ &= \frac{P(x_1, x_2, ..., x_{k-1}, y_k, x_{k+1}, ...x_d)}{P(x_1, x_2, ...x_{k-1}, x_{k+1}, ..., x_d)} \\ &= \frac{P(Y)}{P(x_1, x_2, ...x_{k-1}, x_{k+1}, ..., x_d)} \\ &= \frac{P(Y)P(x_k|x_1, x_2..., x_{k-1}, x_{k+1}, ...x_d)}{P(x_1, x_2, ..., x_{k-1}, x_k, x_{k+1}, ...x_d)} \\ &= \frac{P(Y)P(x_k|x_1, x_2..., x_{k-1}, x_{k+1}, ...x_d)}{P(X)} \\ &= \frac{P(Y)Q(y,x)}{P(X)} \end{aligned}$$

Thus this implies the probability of transitioning $\alpha(x,y) = 1$ from the equations defined above.

This is an important result as in Gibbs sampling always the new state generated is accepted. As we earlier pointed out that Gibbs sampling is useful when we can calculate the conditionals easily. This generally happens when we are sampling from distributions over images where it is easy to compute the conditional probability of a pixel give other pixels.

Chapter 4

Perfect Sampling

The biggest problem with MCMC is we don't know when to stop. In the algorithms discussed above, we had no idea what should be the value of number of iteration to run [3]. There is no surety of convergence. All we know is as $n \to \infty$ we will reach the stationary distribution π but there was no mention of how to check whether it has converged to π. Propp and Wilson [9] showed that this can be done using the methodology we will discuss in this chapter.

4.1 Coupling from the past

Consider, an ergodic (irreducible and apreiodic) Markov chain which has n states and probability of going to state j from state i be p_{ij}. Since the chains are ergodic, if we start at some state, the probability that it ends up at state i, given it ran for long time, is $\pi(i)$ where π is the stationary distribution. To obtain samples, generally, we start with an initial state i and run the Markov chain for some fixed M number of steps. For convenience, we say that the start and finish of the simulation are designated as time -M and time 0.

The aim is to find the state at time 0, given we started with state i. To do this, we start by running the chain from time -1 to time 0. Since the state of the chain at time -1 is determined by the history of the chain from time -M to time -1, that state is unknown to us when we begin our simulation. Hence, we must run the chain from time -1 to time 0 not just once but n times, once for each of the n states of the chain that might occur at time -1.

Let, Markov(i) be a routine which gives a value j with probability p_{ij}. Basically, if we are at state i at time t, the state at t+1 is given by Markov(i). For all times t with $-M \leq t \leq 1$, we can

define a random map f_t , by putting

$$f_t(i) = \text{Markov}(i) \quad \forall i = 1, ..., n$$

using separate calls to Markov() for each time t. We can suppress the details of the construction of the f_t's and imagine that each successive f, is obtained by calling a randomized subroutine Random Map() whose values are actually functions from the state space to itself. The output of the fixed-time simulation is given by $F^0_{-M}(i^0)$, where $F_t 1^t 2$ is defined as the composition $f_{t_2-1} \circ f_{t_2-2} \circ .... \circ f_{t_1}$. However, one need only keep track of the compositions F^0_t, which can be updated via the rule $F^0_t = F^0_{t+1} \circ f_t$. Also note that whenever F^0_t becomes a constant map, with $F^0_t(i) = F^0_t(i') \quad \forall i, i'$, then this will remain true from that point onward, and the value of $F^0_{-M}(i)$ must equal the common value of $F^0_t(i)(1 \leq i \leq n)$, there is no need to go back to time -M once the composed map F^0_t has become a constant map.

When the map F^0_t becomes a constant map, we say *coalescence* occurs from time t to time 0 i.e. coalescence occurs from time t. Backward simulation is the procedure of working backwards until -t is sufficiently large that F^0_t is a constant map, and then values of that occur during the backwards simulation will almost certainly have magnitude much smaller than M. By removing the cutoff M , and running the backwards simulation into the past until F^0_t is constant, we are achieving an output distribution that is equal to the limit, as M goes to infinity, of the output distributions that govern fixed time forward simulation for M steps. However, this limit is equal to π. Hence backwards simulation, with no cut-off, gives a sample whose distribution is governed by the steady-state distribution of the Markov chain. The algorithm is as follows :

Algorithm 2 Coupling from the past

1: $t = 0$
2: $F^0_t * (i) = i \quad \forall i = 1, ..., n$
3: **repeat**
4: $\quad t = t - 1$
5: $\quad f_t = RandomMap()$
6: $\quad F^0_t = F^0_{t+1} \circ f_t$
7: **until** $F^0_t(.)$ is constant
8: **return** the value of F^0_t

It is also proved that with probability 1 the coupling from the past algorithm returns a value and this value is distributed according to the stationary of the Markov chain, which also implies that CFTP algorithm will terminate in finite time and the sample is from stationary distribution i.e. perfect sample [9].

Theorem 4.1.1. *The CFTP algorithm returns a random variable distributed exactly according to the stationary distribution of the Markov chain.*

For proof of the theorem, refer to [9]

4.2 Monotone Monte Carlo

The above algorithm gives us a way to sample from distributions exactly, i.e. we know when to stop. But, if the number of state is too huge, like a binary image of size 256×256 pixels, the number of states are $2^{256\times256} \sim 10^{19730}$ and it is almost impossible to keep track of all states and check the convergence. This motivates the need for another condition on the Markov chain which can help us track all these states easily [8].

Since our procedure uses a random variable to perform the updates at time t, let us define the function $\phi(x, U_t)$, such that $f_t(x) = \phi(x, U_t)$ where U_t is the value of the random variable used at time t to perform the update. Suppose now that the state space S of our Markov chain admits a natural partial ordering $\leq$, and that our update rule ϕ has the property that $x \leq y$ implies $\phi(x, U_0) \leq \phi(y, U_0)$, then we say that our Markov chain gives a monotone Monte Carlo algorithm for approximating the stationary distribution π. From now on assume that S has elements $\hat{0}$ and $\hat{1}$ which represent the minimal and maximal elements of the partial order. That is, $\hat{0} \leq x \leq \hat{1} \quad \forall x \in S$.

The CFTP algorithm discussed in the previous section converges to the point where all states in the state space have coalesced. Since, partial order is maintained in every update, even if we just run the update on the $\hat{0}$ and $\hat{1}$ states, we conclude that Markov chain has coalesced when these two states converges at same value (as this means all possible states in the beginning have coalesced). Every state in between the minimal and maximal state are squeezed in between their values and will be between them after any number of updates. The algorithm is as follows :

Algorithm 3 Montone sequence coupling from the past

1: $t = 1$
2: $upper = \hat{1}$
3: $lower = \hat{0}$
4: **repeat**
5: $\quad upper = \phi_t(upper, u_t)$
6: $\quad lower = \phi_t(lower, u_t)$
7: $\quad t = 2t$
8: **until** $upper = lower$
9: **return** upper

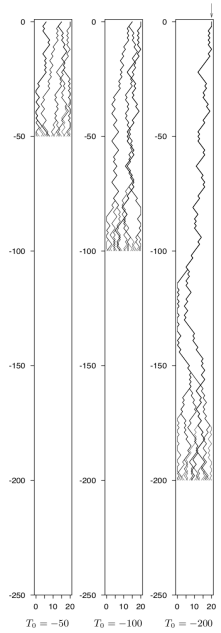


Figure 4.1: *Run of CFTP Algorithm*

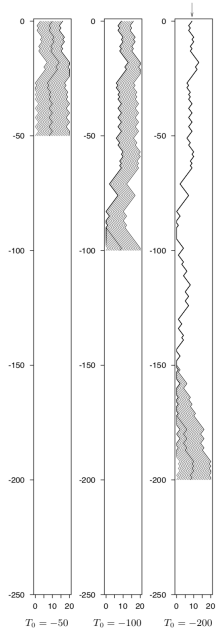


Figure 4.2: *Run of CFTP Algorithm on monotone chains*

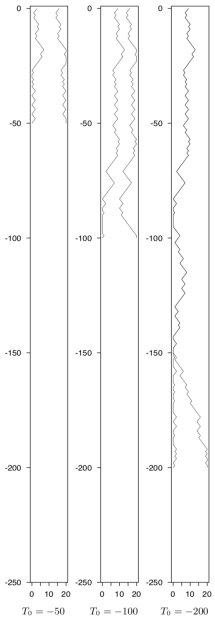


Figure 4.3: *Run of CFTP Algorithm on montone chains with extreme states*

4.3 Perfect Sampling from the Ising Model

Definition. *Ising model consists of discrete variables that represent magnetic dipole moments of atomic spins that can be in one of two states (+1 or -1). The spins are arranged in a graph, usually a lattice, allowing each spin to interact with its neighbors.*

Define a spin system S, on a vertex set V as the set of all ways of assigning a spin $\sigma(i)$("-1" or "+1") to each of the vertices $i \in V$, together with a probability distribution π on the set of such assignment $\sigma(.)$. Define a partial order on the set of configurations by putting $\sigma \geq \tau$ iff $\sigma(i) \geq \tau(i) \quad \forall i \in V$. This, implies the minimal element $\hat{0}$ and maximal element $\hat{1}$ corresponds to $\hat{0}(i) = -1$ and $\hat{1}(i) = +1 \quad \forall i \in V$ respectively. Probability distribution for every assignment, in an assignment is given by,

$$\pi(x) = \frac{e^{\beta J(x)}}{Z_\beta} \qquad \text{where } J(x) = \sum_{i,j} W_{i,j} x(i) x(j)$$

Z_β is the normalization constant which is equal to $\sum_{x \in S} e^{\beta J(x)}$. $W_{i,j}$ is the weight given to the interaction between i and j ($W_{i,j} > 0$). $J(x)$ measures the total interaction of the spin system.

On this model, we can use Gibbs sampling method for perfect sampling when $\beta > 0$. For some given $i \in V$ and configurations σ, τ such that $\sigma(j) \leq \tau(j) \quad \forall j \neq i$. Define $\sigma_{+1}, \sigma_{-1}, \tau_{+1}, \tau_{-1}$

by putting $\sigma_{+1}(i) = +1, \sigma_{+1}(j) = \sigma(j) \quad \forall j \neq i$. Similarly, other configurations are defined. Update on the vertex i of a configuration σ,

$$f_t(\sigma, u_t) = \begin{cases} -1 & u_t < \frac{\pi(\sigma_{-1})}{\pi(\sigma_{-1})+\pi(\sigma_{+1})} \\ +1 & \text{otherwise} \end{cases}$$

where u_t is the random variable distributed uniformly in [0,1].
This preserves the partial order as:
We will show that in every update the partial order is preserved i.e. $\sigma < \tau$. For a given i if $\sigma(j) < \tau(j)$ for all $j \neq i$ then we have,

$$\sum_j W_{i,j}\sigma(j) \leq \sum_j W_{i,j}\tau(j)$$

$$\frac{\pi(\sigma_{-1})}{\pi(\sigma_{+1})} = e^{-2\beta\sum_j W_{i,j}\sigma(j)}$$

$$\frac{\pi(\tau_{-1})}{\pi(\tau_{+1})} = e^{-2\beta\sum_j W_{i,j}\tau(j)}$$

Thus, as $\beta > 0$ we have,

$$\frac{\pi(\sigma_{-1})}{\pi(\sigma_{+1})} \geq \frac{\pi(\tau_{-1})}{\pi(\tau_{+1})} \implies \frac{\pi(\sigma_{-1})}{\pi(\sigma_{+1}) + \pi(\sigma_{-1})} \geq \frac{\pi(\tau_{-1})}{\pi(\tau_{+1}) + \pi(\tau_{-1})}$$

Thus partial order will be maintained and we can use the CFTP algorithm to draw perfect samples from the Ising model.

4.4 Perfect Sampling from Posteriors with Ising Model Prior

We can extend the previous discussion about sampling from the Ising model to sample from posterior distributions whose prior model is the Ising model. We represent the data observed as D and we want to get samples which come from the distribution which is distributed as P(x | D) where x ∈ S. As $P(x|D) = \frac{P(D|x)P(x)}{P(D)}$, $P(D|x)$ is the likelihood term and $P(x)$ is the prior probability. Also, in most of the cases, $P(D|x) = \prod_{i \in V} P(D(i)|x(i))$, here to we assume this is the case. Here too we have the same sample space, but we want to sample from $\phi(x) = P(x|D)$. For given $i \in V$ and configuration σ, τ, such that $\sigma(i) \leq \tau(i) \quad \forall j \neq i$, we define $\sigma_{+1}, \sigma_{-1}, \tau_{+1}, \tau_{-1}$ by putting $\sigma_{+1}(i) = +1, \sigma_{+1}(j) = \sigma(j) \quad \forall j \neq i$. Similarly, other configurations are defined. Update on the vertex i of a configuration σ,

$$f_t(\sigma, u_t) = \begin{cases} -1 & u_t < \frac{\pi(\sigma_{-1})}{\pi(\sigma_{-1})+\pi(\sigma_{+1})} \\ +1 & \text{otherwise} \end{cases}$$

where u_t is the random variable distributed uniformly in [0,1]. This is exactly same as Gibbs sampling procedure. This preserves partial order as :

The only thing that we need to show is $\frac{\phi(\sigma_{-1})}{\phi(\sigma_{+1})} \geq \frac{\phi(\tau_{-1})}{\phi(\tau_{+1})}$ because then we can use the argument given in section 4.3 to show the partial order is maintained.

$$\begin{aligned}
\frac{\phi(\sigma_{-1})}{\phi(\sigma_{+1})} &= \frac{P(\sigma_{-1}|D)}{P(\sigma_{+1}|D)} \\
&= \frac{P(D|\sigma_{-1})P(\sigma_{-1})}{P(D|\sigma_{+1})P(\sigma_{+1})} \\
&= \frac{\prod_{j\in V} P(D(j)|\sigma_{-1}(j))P(\sigma_{-1})}{\prod_{j\in V} P(D(j)|\sigma_{+1}(j))P(\sigma_{+1})} \\
&= \frac{P(D(i)|\sigma_{-1}(i))\pi(\sigma_{-1})}{P(D(i)|\sigma_{+1}(i))\pi(\sigma_{+1})} \\
&= \frac{P(D(i)|-1)\pi(\sigma_{-1})}{P(D(i)|+1)\pi(\sigma_{+1})}
\end{aligned}$$

And we saw that,

$$\begin{aligned}
&\frac{\pi(\sigma_{-1})}{\pi(\sigma_{+1})} \geq \frac{\pi(\tau_{-1})}{\pi(\tau_{+1})} \\
\implies &\frac{P(D(i)|-1)\pi(\sigma_{-1})}{P(D(i)|+1)\pi(\sigma_{+1})} \geq \frac{P(D(i)|-1)\pi(\tau_{-1})}{P(D(i)|+1)\pi(\tau_{+1})} \\
\implies &\frac{\phi(\sigma_{-1})}{\phi(\sigma_{+1})} \geq \frac{\phi(\tau_{-1})}{\phi(\tau_{+1})}
\end{aligned}$$

Hence, the partial order is maintained and we can adapt to a similar sampling procedure as in section 4.3.

4.5 Fills Algorithm

The running time of CFTP algorithm is an unbounded random variable whose order of magnitude is typically unknown a priori and which is not independent of the state sampled, so a naive user with limited patience who aborts a long run of the algorithm will introduce bias [5].

Let P be an ergodic transition matrix with monotone time-reversal $\bar{P}$ on a poset S possessing a unique minimum element $\hat{0}$ and a unique maximum element $\hat{1}$. The algorithm is as shown in Algorithm 8.

The above algorithm is for general Markov chains which need not be monotone in nature. As described in last section we can use this algorithm for monotone chains by only starting at maximal state and minimal state. By the similar argument given above, if these two chains coalesce then we found a sample from the stationary distribution π. The claim of unbiasedness with respect to user impatience follows from the very nature of rejection sampling together with the fact that all information is erased after each iteration.

Algorithm 4 Fills algorithm

$t = 1$
repeat
  $x_t = z$
  Generate $X_{t-1}|x_t, X_{t-2}|x_{t-1}, ..., X_0|x_1$
  Generate $[U_1|x_0, x_1], [U_2|x_1, x_2], ..., [U_t|x_{t-1}, x_t]$
  Begin chains $X^{0,1}, X^{0,2}, ...X^{0,k}$ in all possible initial states at time 0
  Use the common $U_1, U_2, ..., U_t$to update chains
  $t = 2t$
until Until chains coalesce
return x_t

Theorem 4.5.1. *The Fills algorithm returns a random variable distributed exactly according to the stationary distribution of the markov chain.*

Proof.

$$P[X_0 = x|accept] = \frac{P[z \to x]P[C_T(z)|x \to z]}{\sum_{x'} P[z \to x']P[C_T(z)|x' \to z]}$$

As coalescence event entails each $x' \to z$, we have $\forall x'$,

$$P[C_T(z)|x' \to z] = \frac{P[C_t(z) \text{ and } x' \to z]}{P[x' \to z]} = \frac{P[C_t(z)]}{P[x' \to z]}$$

From the detailed balance condition $\pi(x)P(x \to z) = \pi(z)P(z \to x)$

$$\begin{aligned} P[X_0 = x|accept] &= \frac{P[z \to x]/P[x \to z]}{\sum_{x'} P[z \to x']/P[x' \to z]} \\ &= \frac{\pi(x)/\pi(z)}{\sum_{x'} \pi(x')/\pi(z)} = \pi(x) \end{aligned}$$

□

4.6 Perfect Sampling from General Systems

4.6.1 Bounding Chains

Bounding chains [7] are useful for MCMC in three ways. First, they are formed from a natural extension of couplings known as complete couplings. Second, bounding chains are themselves Markov chains that can be simulated. Third, bounding chains allow a user to utilize perfect sampling algorithms such as the coupling from the past.

Definition. *We say that M′ is a bounding chain for M if there exists a coupling between M′ and M such that*

$$X_t(v) \in Y_t(v) \quad \forall v \implies X_{t+1}(v) \in Y_{t+1}(v) \quad \forall v$$

The Potts model is an extension of the Ising model from statistical physics. Each node of a graph is assigned a color from $\{1, 2, ...k\}$ in a configuration. The energy of the configuration x is $H(x) = -\sum_v \sum_{w \in N_v} 1_{x(v)==x(w)}$ where N_v is set of nodes adjacent to v. The probability of choosing a particular configuration is

$$\pi(x) = \frac{1}{Z} \exp\{-\beta J H(x)\}$$

where J is either 1 (attractive system) or -1 (repelling system), and Z is known as partition function. The algorithm is as follows (for general weighted combination of neighbours):

Algorithm 5 Bounding chain for Gibbs sampler

1: Choose $v \in_U \{1, 2, ...n\}$, Let N_v be the neighbours of v
2: Let $y(v) \leftarrow \phi$
3: **repeat**
4: Choose $c \in_U \{1, 2, ..., k\}$
5: Choose $u \in_U [0, 1]$
6: Let b_c be the w neighbouring v with y(w) = {c}
7: Let d_c be the w neighbouring v with c ∈ y(w)
8: **if** $u < \gamma^{f_{max}(b_c, d_c)}$ **then**
9: Let $y(v) \leftarrow y(v) \cup \{c\}$
10: **until** $u \leq \gamma^{f_{min}(d_c, b_c)}$ or $|y(v)| > \Delta$

Theorem 4.6.1. *Let τ be the first time that the bounding chain detects complete coupling. Then when T satisfies* $\exp(\frac{2}{T}) < \frac{k\Delta}{k\Delta - 1}$, *there exist a constant $\beta \in (0, 1)$ such that*

$$P(\tau \geq (-\ln \beta) n \ln n + \theta) \leq \beta^{\theta}$$

Proof. Let W_t denotes the number of nodes v with $|Y_t(v)| \geq 1$. We begin at $W_0 = n$ and wish to find the expected time until $W_t = 0$. Given Y_t we wish to find the expected value of W_{t+1}.
If at time $t + 1$ we select a node with $|Y_t| > 1$, then it would decrease W_{t+1} by 1 at that point.
Now, if we select a color c for the node v that lies in $Y_t(w)$ for some neighbour w of v and we

don't exit of the repeat loop, then it increases W_{t+1} by 1.

$$\begin{aligned}
E[W_{t+1}|Y_t] &\leq W_t - \sum_v \frac{\mathbb{1}_{|Y(v)|>1}}{n} + \sum_v \frac{1}{n} \sum_{w \in N_v} \mathbb{1}_{|Y_t(w)>1} |Y_t(w)|(1-\gamma^{-1}) \\
&\leq W_t - \frac{W_t}{n} + \frac{k}{n}(1-\gamma^{-1}) \sum_v \sum_{w \in N_v} \mathbb{1}_{|Y_t(w)>1} \\
&\leq W_t - \frac{W_t}{n} + \frac{k}{n}(1-\gamma^{-1}) W_t \Delta \\
&\leq \beta W_t = \beta E[W_t]
\end{aligned}$$

where $\beta = \left[1 - \frac{1}{n}\left(1 - k(1-\gamma^{-1})\Delta\right)\right]$, and thus it gives $E[W_t] \leq \beta^t n$. Also, $E[W_t] \to 0$ implies $\beta < 1$ which gives $\gamma < \frac{k\Delta}{k\Delta - 1}$. Application of Markov's inequality gives us $P(W_t > 1) \leq E[W_t] \leq \beta^t n$ and the result directly follows. □

We now extend the discussion of perfect sampling using bounding chains to sample from posteriors with an i.i.d likelihood model which is common in many image processing scenarios. Assume that the observed data is given by Z such that $P(X|Z) \propto P(Z|X).P(X)$ where X is distributed as given by the Potts model and $P(Z|X) = \prod_{i \in V} P(Z_i|X_i)$. The Markov chain update using the Gibbs sampler can be written as,

Algorithm 6 Gibbs sampler with acceptance rejection

1: Choose $X_i \in_U \{1, 2, ...n\}$, Let N_{X_i} be the neighbours of X_i
2: **repeat**
3: Choose $c \in_U \{1, 2, ...k\}$
4: Choose $u \in_U [0, 1]$
5: **until** $u \leq \frac{e^{\beta J(x_i)} P(Z_i|X_i)}{\sum_c e^{\beta J(x_i)} P(Z_i|X_i)}$
6: Assign color c to X_i

We will now go over why the acceptance rejection chain for the above sampling method is correct. To be able to do this, we notice that the above is a step of Gibbs sampling which should be able to sample from $P(X_i|X_{\ i}, Z)$.

The probability of assigning a color c to X should be given by $P(X = c|X_i, Z)$

$$P(X_i = c|X_{\sim i}, Z) = \sum_j P(\text{we accept in the jth step and the color chosen} = c)$$

$$= \sum_j P(\text{color chosen} = c) * P(\text{we accept c in the jth step}) * P(\text{first accept in the jth step})$$

$$= \sum_j \frac{1}{|C|} * P(u < \frac{e^{\beta J(c)} * P(Z_i|X_i=c)}{A}) * (\sum_{c' \in C} P(\text{color c' was chosen and rejected }))^{j-1}$$

$$= \sum_j \frac{1}{|C|} * \frac{e^{\beta J(c)} * P(Z_i|X_i=c)}{A} * \left(\sum_{c' \in C} \frac{1}{|C|} * \left(1 - \frac{e^{\beta J(c')} * P(Z_i|X_i=c')}{A}\right)\right)^{j-1}$$

$$= \frac{1}{|C|} * \frac{\frac{e^{\beta J(c)} * P(Z_i|X_i=c)}{A}}{1 - \sum_{c' \in C} \frac{1}{|C|} * (1 - \frac{e^{\beta J(c')} * P(Z_i|X_i=c')}{A})}$$

$$= \frac{1}{|C|} * \frac{\frac{e^{\beta J(c)} * P(Z_i|X_i=c)}{A}}{\frac{1}{|C|} * \sum_{c' \in C} (\frac{e^{\beta J(c')} * P(Z_i|X_i=c')}{A})}$$

$$= \frac{e^{\beta J(c)} * P(Z_i|X_i=c')}{\sum_{c' \in C} (e^{\beta J(c')} * P(Z_i|X_i=c'))}$$

$$= \frac{P(X_i=c|X_{\sim i}) * P(Z_i|X_i=c)}{\sum_{c' \in C} P(X_i=c'|X_{\sim i}) * P(Z_i|X_i=c')}$$

This means that we are indeed sampling from the posterior distribution and therefore the algorithm is correct. The A term is chosen such that it remains a probability distribution and we could get the best possible speedup. The above proof could have easily been done taking into consideration just the prior term and therefore the Markov chain described for the prior/posterior ,indeed ,samples from the required distribution.

Bounding chains helps to detect convergence of a chain, but it does not define the stopping point and so if we stop at early point and output the sample, we might get biased samples. So we do a perfect sampling procedure above this to get unbiased or "perfect" samples using CFTP or Fills Algorithm.

Bounding chains helps in maintaining and tracking the states of all chains. Now if they are combined with the perfect sampling algorithms, then they can yield perfect samples from any distribution which allows sampling from the conditional. We propose novel algorithm to sample from the given distribution which need not be monotonic using CFTP over Bounding chains algorithm. In CFTP, if the states have not converged, we go back in time and run the sampling procedure again using the same random numbers at time points we have already seen. nce all the states have coalesced to a single state, we know that convergence has occurred. This coalesced is now detected using the book keeping of the bounding chains as described in the algorithm below.

Algorithm 7 CFTP with bounding chains

$T = 1$
repeat
  $Y(v) = \mathcal{C} \quad \forall v \in \{1, 2..., n\}$
  for $t = 1 : T$ **do**
    if $t < T/2$ **then**
      $U_t \leftarrow$ Random numbers used in previous t^{th} run
    else
      $U_t \leftarrow U[0,1]^V$
    for each $v \in \{1, 2, ...n\}$ **do**
      Let N_v be the neighbours of v
      Let $Y(v) \leftarrow \phi$
      repeat
        Choose $c \in_U \{1, 2, ..., k\}$
        Choose $u \in_U U_t$
        Let b_c be the w neighbouring v with $Y(w) = \{c\}$
        Let d_c be the w neighbouring v with c $\in Y(w)$
        if $u < \gamma^{f_{max}(b_c, d_c)}$ **then**
          Let $Y(v) \leftarrow Y(v) \cup \{c\}$
      until $u \leq \gamma^{f_{min}(d_c, b_c)}$ or $|Y(v)| > \Delta$
  $T = 2T$
until $|Y(v)| = 1 \quad \forall v \in \{1, 2...., n\}$
return x_T s.t if $c \in Y(v)$ then $x_T(v) = c$

But, the running time of CFTP algorithm is an unbounded random variable whose order of magnitude is typically unknown a priori and which is not independent of the state sampled A naive user with limited patience who aborts a long run of the algorithm will introduce bias. Claim of unbiasedness in Fills with respect to user impatience follows from the very nature of acceptance rejection sampling together with the fact that all information is erased after each iteration. Thus, if CFTP is always run for shorter time we will always generate biased samples in the sense that some part of the distribution will never be visited.

Fills algorithm solves this problem by simulating a single chain from arbitrary initial state and simulating all possible chains back in time in synchrony with the chain simulated in the forward direction. Fills algorithm simulates reversed Markov chains using a updating function ϕ s.t. $x^{(t+1)} = \phi(x^{(t)}, R^{(t-1)})$. For general chains we propose a method to run all chains in synchrony with the forward simulated chain in bounding chains algorithm.

Algorithm 8 Fills algorithm with bounding chains

$T = 1$
repeat
  $Y(v) = \mathcal{C} \quad \forall v \in \{1, 2..., n\}$
  $x_T = z \in_U \{1, 2..., k\}^V$
  Generate $X_{T-1}|x_T, X_{T-2}|x_{T-1}, ..., X_0|x_1$
  Generate $[U_1|x_0 \to x_1], [U_2|x_1 \to x_2], ..., [U_T|x_{T-1} \to x_T]$
  for $t = 1 : T$ **do**
    for each $v \in \{1, 2, ...n\}$ **do**
      Let N_v be the neighbours of v
      Let $Y(v) \leftarrow \phi$
      repeat
        Choose $c \in_U \{1, 2, ..., k\}$
        Choose $u \in_U U_t^c$
        Let b_c be the w neighbouring v with $Y(w) = \{c\}$
        Let d_c be the w neighbouring v with c $\in Y(w)$
        if $u < \gamma^{f_{max}(b_c, d_c)}$ **then**
          Let $Y(v) \leftarrow Y(v) \cup \{c\}$
      until $u \leq \gamma^{f_{min}(d_c, b_c)}$ or $|Y(v)| > \Delta$
  $T = 2T$
until $|Y(v)| = 1 \quad \forall v \in \{1, 2...., n\}$
return x_0

Fills algorithm for generation of perfect samples can be interrupted at any time, and yet, the samples generated are not biased because, the running time of the algorithm and the returned values are independent. Thus, Fill's algorithm is similar to CFTP in that it uses Markov chains to produce perfect samples, but it is based on the idea of rejection sampling instead of the concept of coupling.

Chapter 5

Experiments and Results

5.1 Various MCMC sampling methods

We wanted to check and compare various Markov chain sampling methods. For this purpose, we sampled images for different temperatures i.e. for different values β's and compared the results from different methods. We also implemented Perfect Sampling using Bounding chains. The results for attractive and repelling spins systems are as shown:

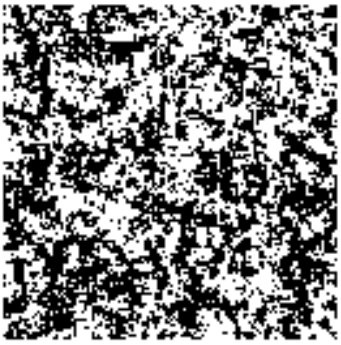


Figure 5.1: *Bounding Chains with Attractive spin system and β is 1/3*

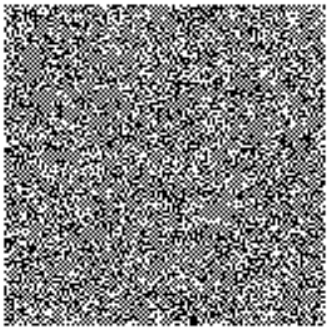
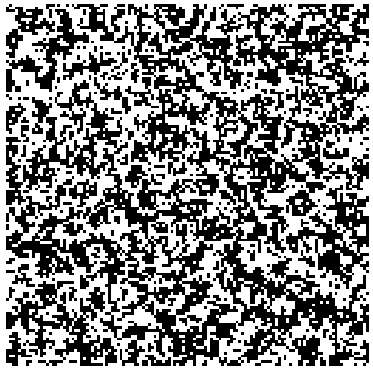


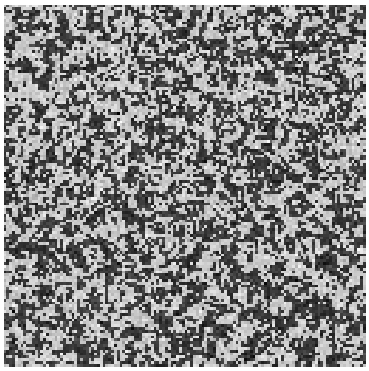
Figure 5.2: *Bounding Chains with Repelling spin system and β is 1/3*

5.2 Perfect Sampling from posteriors

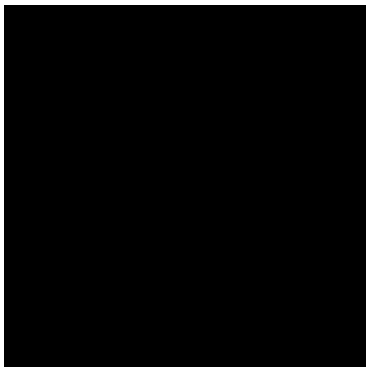
We wanted to show how good perfect sampling is and to do so, we generated a random binary (bi-label) sample from the prior using the Ising model using clique of size 2 only i.e. using only neighbourhood interactions. From this we generate various test images by sampling from two different Gaussian depending upon the pixel label. Now, using this as given data we generate samples from posterior distribution and take its average which is MAP estimate. The results are as shown :



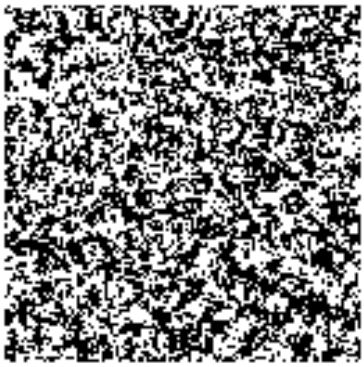
(a) *Prior Image*



(b) *Data Image*

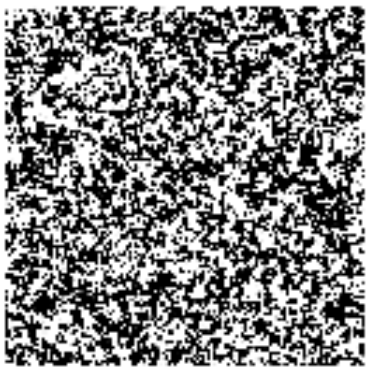


(c) *Variance Image*

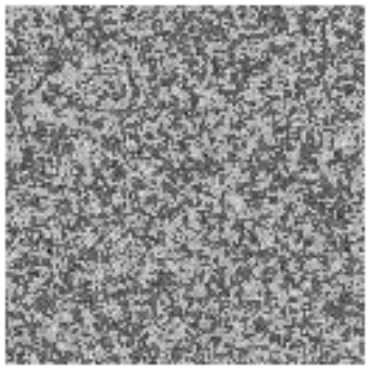


(d) *Average Posterior Sample*

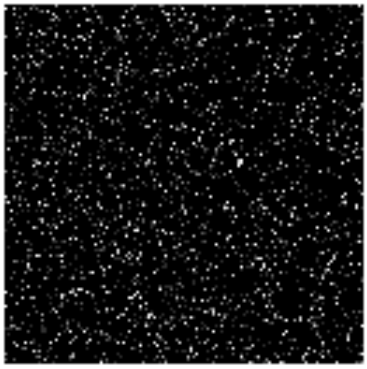
Figure 5.3: *Image Samples when μ_1 = 10, μ_2 = -10, σ_1 = 2, and σ_2 = 2. Number of samples generated 20. Total Error is 0. (Image size is* 128×128*)*



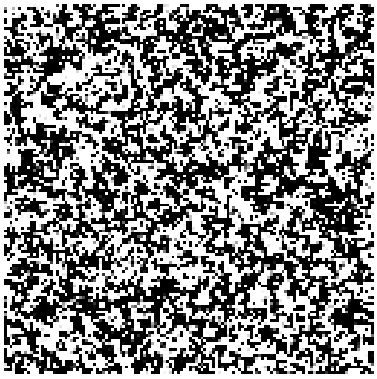
(a) *Prior Image*



(b) *Data Image*



(c) *Variance Image*



(d) *Average Posterior Sample*

Figure 5.4: *Image Samples when μ_1 = 10, μ_2 = -10, σ_1 = 5, and σ_2 = 5. Number of samples generated 20. Total Error is 556. (Image size is* 128×128*)*

5.3 Expectation Maximization using Sampling on MRI Image

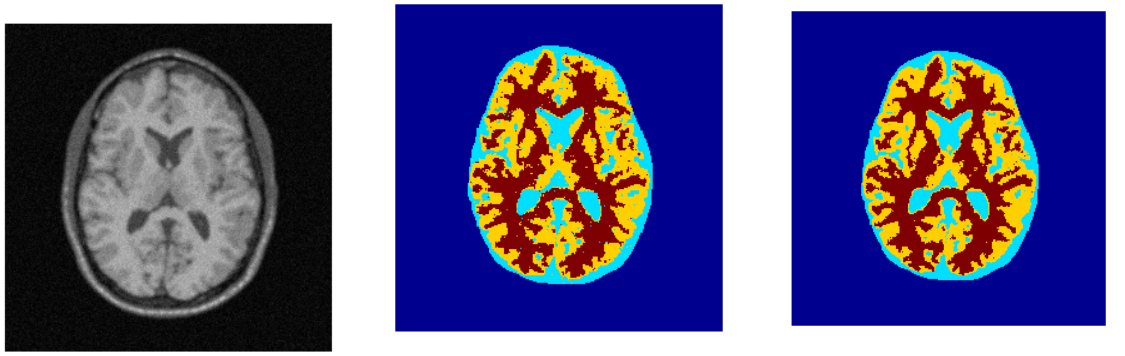
We show the practical application of posterior sampling on images. In the problem of soft image segmentation with MRF prior on label image and Gaussian mixture mode likelihood, the E step expectation is analytically intractable. There are algorithms which uses an approximation in E step which approximates it to a distribution whose expectation is easy to obtain. Instead of approximating the E-step with an approximate distribution whose expectation is easy to obtain, we retain the original distribution and perform perfect sampling on this distribution

$$E_{P(x_i,x_{\sim i}|y,\theta^t)}[\log P(y_i|x_i,\theta)] \approx E_{P(x_i|x_{\sim i}y,\theta^t)}[\log P(y_i|x_i,\theta)]$$

We estimate the expectation using the average of a few samples that are obtained using the perfect sampling procedure.

$$E_{P(x_i,x_{\sim i}|y,\theta^t)}[\log P(y_i|x_i,\theta)] \approx \frac{1}{\mathcal{S}} \sum_{\substack{s=1 \\ x^s \sim P(x|y,\theta^t)}}^{\mathcal{S}} \log P(y_i|x_i^s,\theta)$$

The M step is same as the original method. We got the following results:



(a) *Original Brain MRI Image*

(b) *Segmented Brain MRI using approximations in E step*

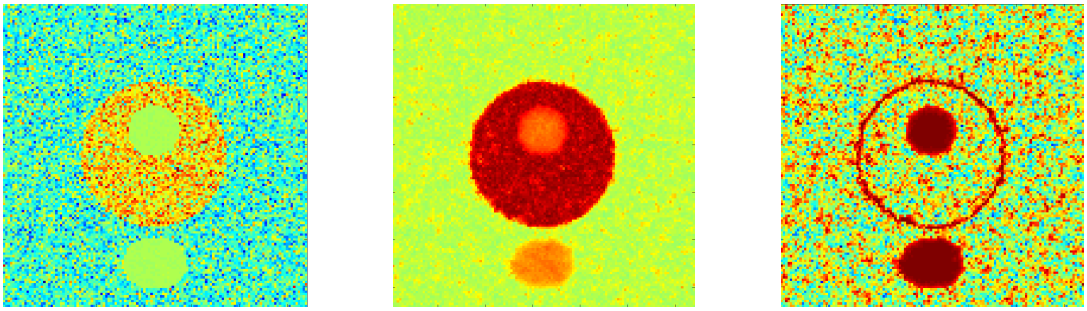
(c) *Segmented Brain MRI Image without approximations in E step*

Figure 5.5: *Segmentation on Brain MRI*

5.4 Uncertainty estimation using perfect sampling

In most of today's segmentation problems the uncertainty estimations are not clear. We want to identify the regions where the segmentation is sure and where it is not. Thus, their is need for perfect sampling from the estimated posterior distribution for uncertainty estimation in various real life scenarios like lesion detection. We can not use Gibbs sampling because Insufficient burn-in artificially inflates the variance and hence we get incorrect uncertainty estimations. If we try to

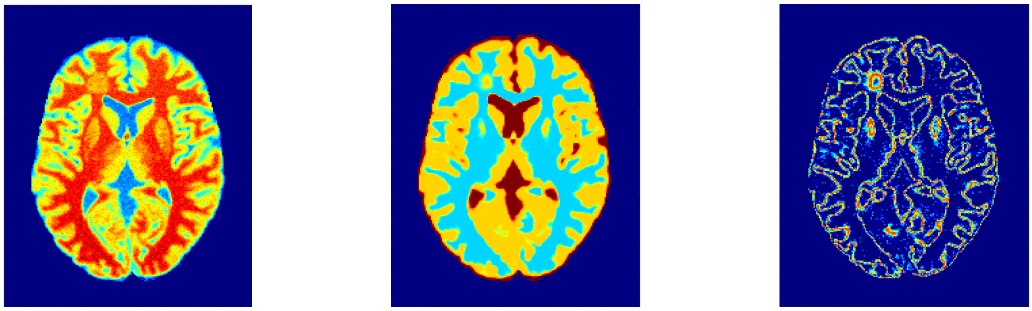
conservatively estimate the burn-in for Gibbs sampling then it can make the burn-in iterations too large making the sampling too slow. Hence we use perfect sampling algorithms over bounding chains to sample from arbitrary posteriors as described before. To evaluate the efficacy of perfect sampling we first tested it on simulated lesion data and the results are as shown: The simulated



(a) *Original Image* **(b)** *Mean Label Image* **(c)** *Variance Image*

Figure 5.6: *Uncertainty Estimation over samples from estimated posterior on simulation of lesion*

data is a circular patch with guassian noise (0 mean and 1.414 standard deviation) added on top of it. We further add two lesion regions in the foreground and background to evaluate efficacy of proposed perfect sampling algorithm for uncertainty estimation. We can clearly see that for segmentation with 2 labels, the uncertainty in lesion region is very high, where intensity is almost in the middle range of two labels. Also the boundary points are regions of high uncertainty because of smoothing. We also evaluated the performance of perfect sampling on simulated Brain MRI data. We use EM algorithm with MRF prior as discussed above with approximation in E-step to segment the brain image into 3 segments (CSF + Gray Matter + White Matter). We took a slice with visible subcortical structure and added spatially varying smoothing, bias field, lesion and gaussian noise to it. The results of uncertainty estimations are as shown:



(a) *MRI with noise and bias* **(b)** *Mean Label Image* **(c)** *Variance Image*

Figure 5.7: *Uncertainty Estimation over samples from estimated posterior on Brain slice with subcortical structure and simulated lesion*

We can clearly see high uncertainty in the region of thalamus and in the lesion. This is because it is very difficult to segment these regions into one of the three classes owing to the intensity values in these regions.

Chapter 6

Conclusion and Further Reading

We have discussed various methods of sampling and there applicability, also how they are useful in context of images. Later, we introduced the idea of perfect sampling and how perfect sampling methods are in general better, as they provide the idea when to stop the algorithm i.e. when it has converged to the stationary distribution. In perfect sampling, we specifically studied Coupling from the past (CFTP) and Fills algorithm which aims at removing user impatient bias from CFTP. We have studied perfect sampling from Ising model, how it extends for sampling over posteriors, method of bounding chains for general Potts model and how it is used for perfect sampling.

We studied another MCMC method Swendsen-Wang which samples from Potts model, we would like to implement it in the future. We would also like to formulate perfect sampling procedure for the CFTP for the Swendsen-Wang algorithm. In the whole report we only talked about the sampling in discrete space Markov chains, we will try to understand methods for sampling in continues space Markov chain. We also want to check the efficacy of the approach of perfect sampling on various other real data sets for Infant MRI Segmentation and Tumor Segmentation.

Bibliography

[1] Song-Chun Zhu A. Barbu. Generalizing swendsen-wang to sampling arbitrary posterior probabilities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2005.

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

[3] George Casella, Michael Lavine, and Christian P Robert. Explaining the perfect sampler. *The American Statistician*, 55(4):299–305, 2001.

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

[5] James Allen Fill. An interruptible algorithm for perfect sampling via markov chains. In *Proceedings of the Twenty-ninth Annual ACM Symposium on Theory of Computing*, STOC '97, pages 688–695, New York, NY, USA, 1997. ACM.

[6] Mark Huber. A bounding chain for swendsen wang. *Random Structures Algorithms*, 22(1):43–59, 1 2003.

[7] Mark Huber. Perfect sampling using bounding chains. *Annals of Applied Probability*, pages 734–753, 2004.

[8] David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

[9] James Gary Propp and David Bruce Wilson. Exact sampling with coupled markov chains and applications to statistical mechanics. *Random structures and Algorithms*, 9(1-2):223–252, 1996.

[10] Yongyue Zhang, Michael Brady, and Stephen Smith. Segmentation of brain mr images through a hidden markov random field model and the expectation-maximization algorithm. *IEEE transactions on medical imaging*, 20(1):45–57, 2001.